UNIVERSIDAD DE SALAMANCA

GRADO EN ESTADÍSTICA

FACULTAD DE CIENCIAS

INSTITUTO DE INVESTIGACIÓN BIOMÉDICA

DE SALAMANCA

Trabajo de Fin de Grado: Anexo

Predicción de eventos cardiovasculares y hemorrágicos en pacientes con doble antiagregación con modelos Machine Learning

Prediction of cardiovascular and bleeding events in patients with double antiaggregation with machine learning models

Autor: **Pablo Pérez Sánchez**

Tutores: Dr. José Manuel Sánchez Santos

Dr. Pedro Luis Sánchez Fernández

Salamanca, 2020

UNIVERSIDAD DE SALAMANCA

GRADO EN ESTADÍSTICA

FACULTAD DE CIENCIAS

INSTITUTO DE INVESTIGACIÓN BIOMÉDICA DE SALAMANCA

Trabajo de Fin de Grado: Anexo

Predicción de eventos cardiovasculares y hemorrágicos en pacientes con doble antiagregación con modelos Machine Learning

Prediction of cardiovascular and bleeding events in patients with double antiaggregation with machine learning models

Tutores:

Dr. José Manuel Sánchez Santos

Dr. Pedro Luis Sánchez Fernández

Autor: Pablo Pérez Sánchez

Salamanca, 2020

INDICE

1. Definiciones 3
2. Gráficos y tablas 6
3. Código 18
0_ExploreDatabase 18
1_PreprocessDatabase 30
2_Model 36

1. Definiciones

- **Acceso femoral.**
  Lugar de acceso temporal en la hemodiálisis, se lleva acabo al realizar la intervención percutánea. Este acceso se localiza entre la espina iliaca anterosuperior y la sínfisis del pubis.

- **Accidente cerebrovascular.**
  El accidente cerebrovascular o derrame cerebral se produce cuando se detiene el flujo de sangre a una parte del cerebro.

- **ACEI_ARB.**
  Son un tipo de medicamento que sirve para tratar la elevación de la presión arterial.

- **Angina Inestable.**
  La angina de pecho inestable es una afección en la cual el corazón no recibe suficiente flujo de sangre y oxígeno. Es un signo de alto riesgo de infarto de miocardio o muerte súbita. Se caracteriza por molestia en el tórax o los brazos. Surge durante reposo, es intensa o va en crescendo (más intensa, duradera o frecuente).

- **Anticoagulantes orales.**
  Sí el paciente ha recibido un tratamiento vía oral.

- **Baipás coronario (anterior).**
  El Baipás coronario es un injerto, derivación aortocoronaria o revascularización miocárdica quirúrgica. Se utiliza para mejorar el riego sanguíneo al miocardio en pacientes con obstrucción de las arterias coronarias.

- **Betabloqueantes.**
  Los betabloqueantes son un tipo de medicamentos tratan la presión arterial alta y otras afecciones, como problemas cardíacos.

- **Cáncer.**
  El cáncer se puede originar en cualquier parte del cuerpo. Comienza cuando las células crecen descontroladamente sobrepasando a las células normales, lo cual dificulta que el cuerpo funcione de la manera que debería.

- **Clopidogrel.**
  Medicamento antiagregante que forma parte del tratamiento de doble antiagregación.

- **Creatinina al ingreso.**
  La cantidad de creatinina que se le administra al paciente cuando se realiza el ingreso.

- **Diabetes mellitus.**
  La diabetes mellitus es una enfermedad en la que el páncreas no elabora o produce muy poca insulina o las células del cuerpo no reaccionan a la insulina que se produce.

- **Dislipemia.**
  La dislipemia es una afección en la cual se produce una alteración d ellos niveles de lípidos en sangre.

- **Edad.**
  Edad que tiene los pacientes.

- **Enfermedad vascular periférica.**
  La enfermedad vascular periférica es la obstrucción de los vasos sanguíneos fuera del corazón.

- **Estatina.**
  La estatina es un medicamento para bajar el colesterol.

- **Fallecimiento.**
  Marca la muerte o no del paciente.

- **Hemoglobina al ingreso.**
  La cantidad de hemoglobina que se le administra al paciente cuando se realiza el ingreso.

- **Hemorragia.**
  Si se le produjo una hemorragia al paciente durante su seguimiento.

- **Hemorragia (anterior).**
  Se refiere a hemorragias producidas con anterioridad al registro del paciente**.**

- **Hemorragia en el Hospital.**
  Si se produjo una hemorragia en el hospital durante los días que estuvo ingresado.

- **Hipertensión.**
  La hipertensión arterial es una patología crónica en la que los vasos sanguíneos tienen una tensión constantemente alta.

- **Infarto agudo de miocardio (anterior).**
  El infarto agudo de miocardio (IAM) es una enfermedad que ocurre como consecuencia de la obstrucción de una arteria coronaria por un trombo, disminuyendo la cantidad de oxígeno que llega a las células. Se refiere a infartos producidos con anterioridad al registro del paciente.

- **Infarto agudo de miocardio con elevación del ST.**
  Infarto agudo de miocardio que se produce con elevación del segmento ST (SCACEST), este tipo de infartos suelen ser más graves que los IAM normales.

- **Intervención percutánea coronaria (anterior).**
  Esta intervención trata de pasar un catéter a través de una arteria femoral o radial, de esta forma logran alcanzar el trombo en las arterias coronarias con el catéter y lo abren mecánicamente. Se refiere a intervenciones producidas con anterioridad al registro del paciente.

- **Killip2orMore.**
  Killp-Kimball es una clasificación que permite establecer pronóstico de la evolución de la afección, y las probabilidades de muerte en los 30 primeros días tras el infarto. Tiene 4 escalas, siendo 1 la más leve hasta 4 la más grave.

- **Multivessel.**
  Enfermedad multivaso se refiere a la afección de más de un vaso al sufrir un infarto.

- **No stent.**
  No se le implantó stent al paciente en la intervención percutánea.

- **Prasugrel.**
  Medicamento antiagregante que forma parte del tratamiento de doble antiagregación.

- **Reinfarto.**
  Si se produjo un reinfarto al paciente durante su seguimiento.

- **Reinfarto en el hospital.**
  Si se produjo un reinfarto en el hospital durante los días que estuvo ingresado.

- **Revascularización completa.**
  Una revascularización completa se realiza cuando nos encontramos con una enfermedad coronaria múltiple. Se trata de arreglar todas las cardiopatías.

- **Sexo**
  Sexo al que pertenece el paciente.

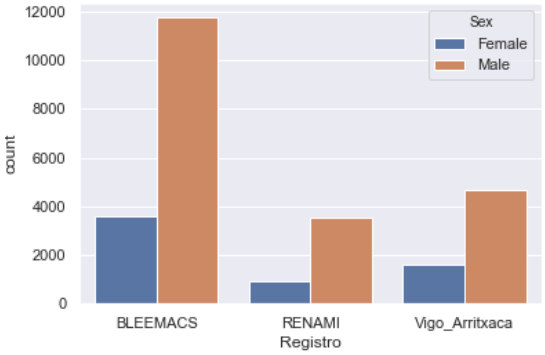
- **Stent farmacológico.**
  Un stent farmacológico es un tubo pequeño que se localiza en la arteria coronaria, el uso de un stent farmacológico es mejor al uso de un stent metálico dado que logra reducir la trombosis.

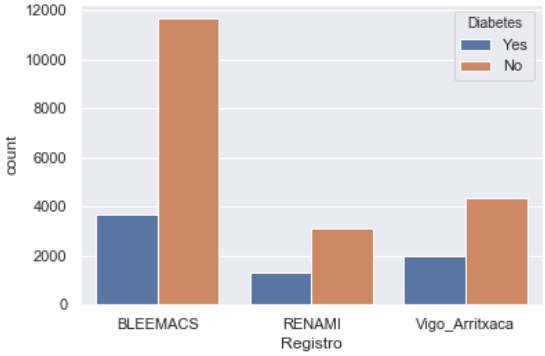
- **Ticagrelor.**
  Medicamento antiagregante que forma parte del tratamiento de doble antiagregación.

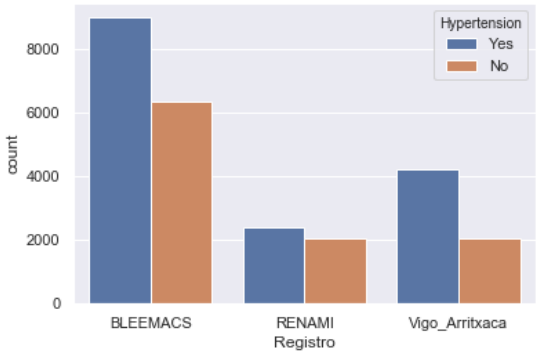
- **Transfusión.**
  Si se le tuvo que realizar transfusión al paciente durante su seguimiento.

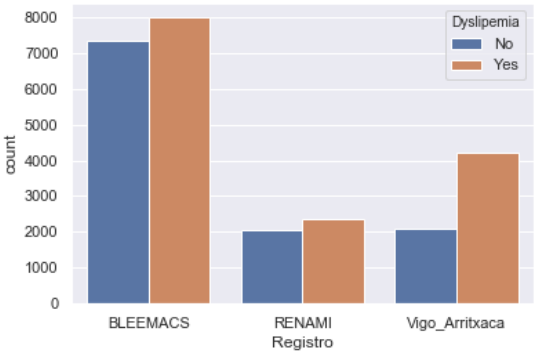
- **Transfusión IH.**
  Se realizo transfusión de IH en el momento del ingreso del paciente.

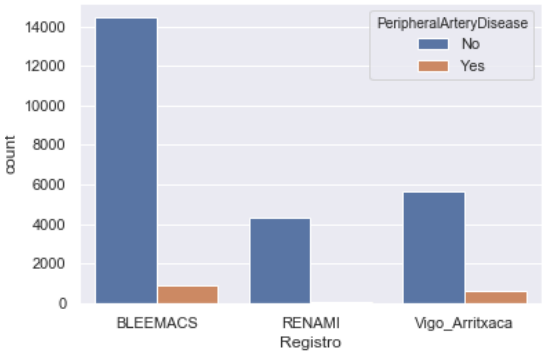
2. Gráficos y tablas

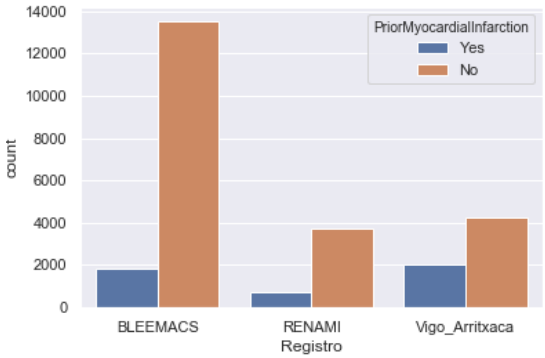


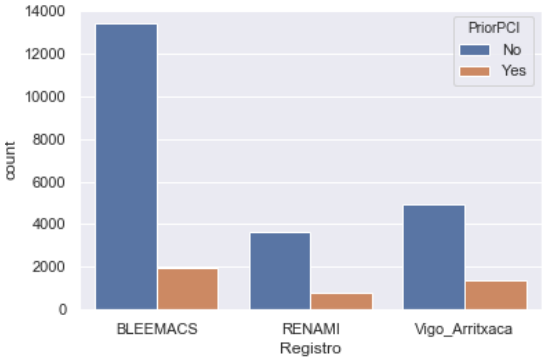


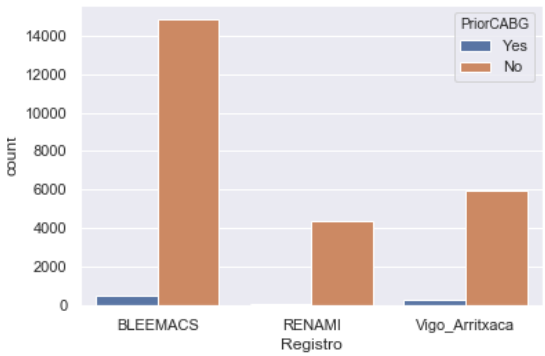


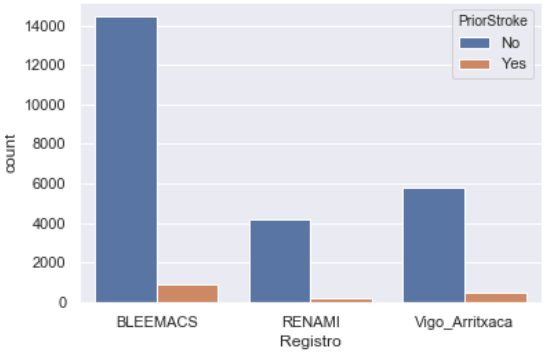


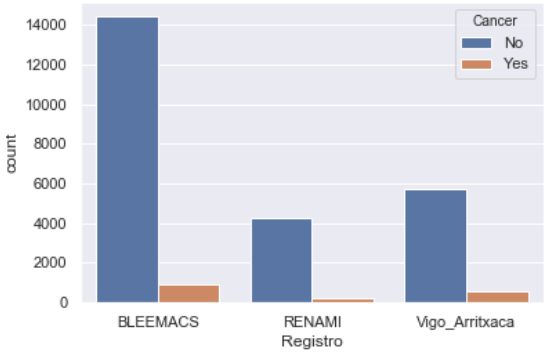


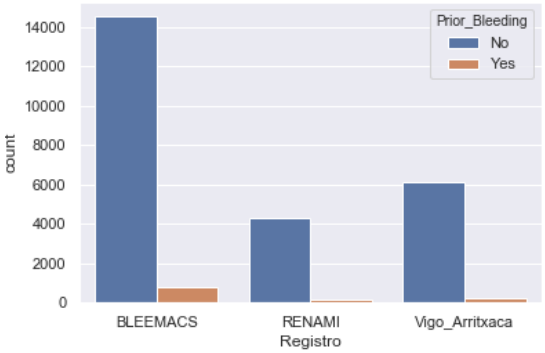


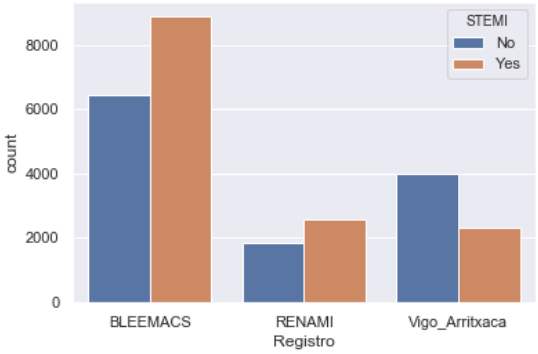


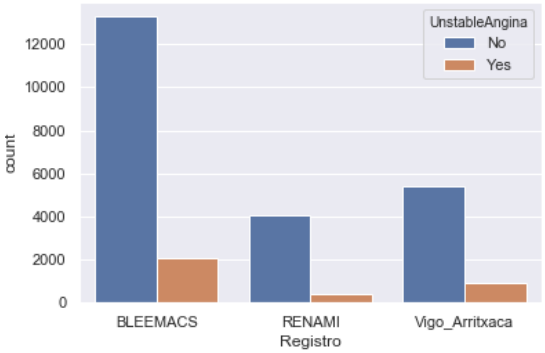


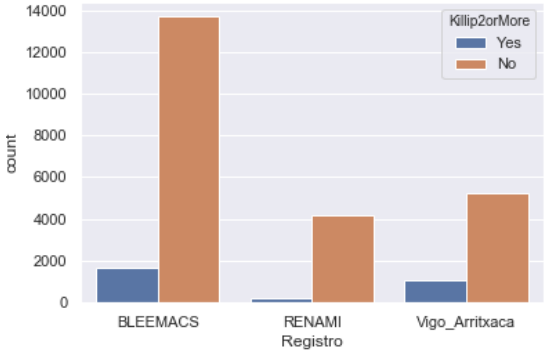

PriorStroke
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

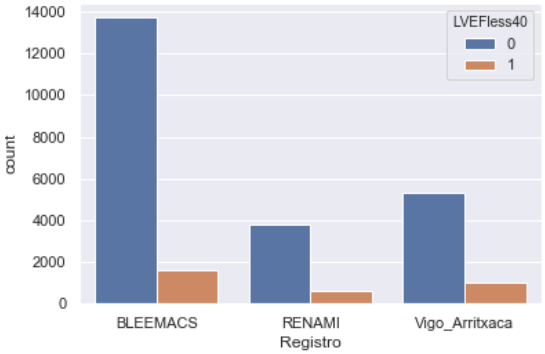

Cancer
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

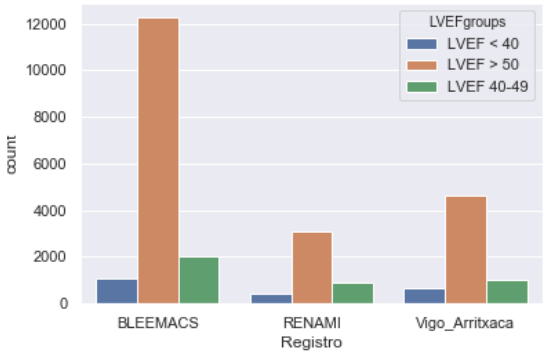

Prior_Bleeding
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

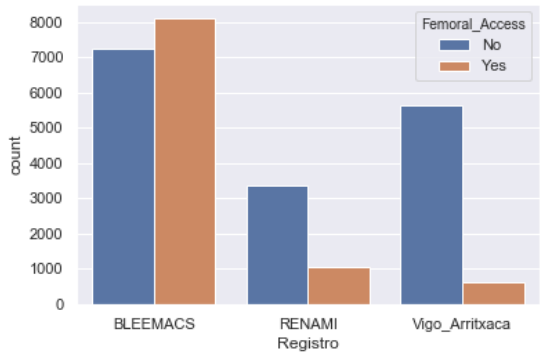

STEMI
No
Yes
count
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

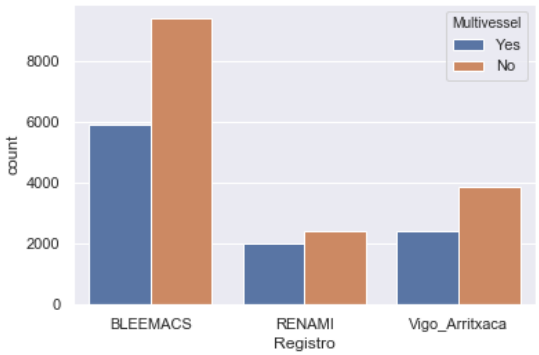

UnstableAngina
No
Yes
count
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

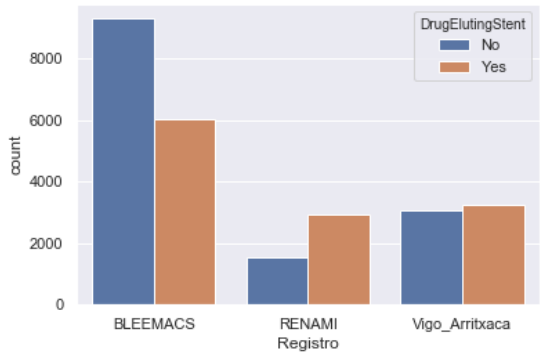

Killip2orMore
Yes
No
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

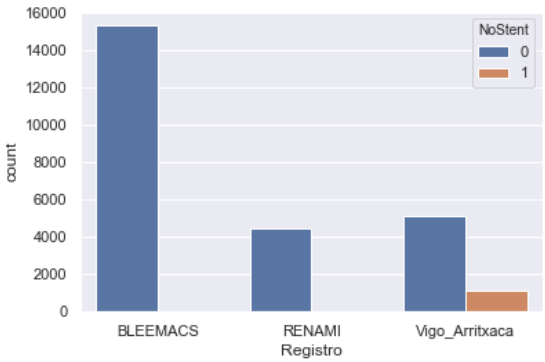

LVEFless40
0
1
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

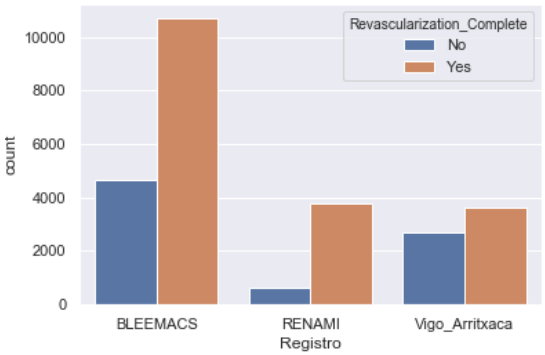

LVEFgroups
LVEF < 40
LVEF > 50
LVEF 40-49
count
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

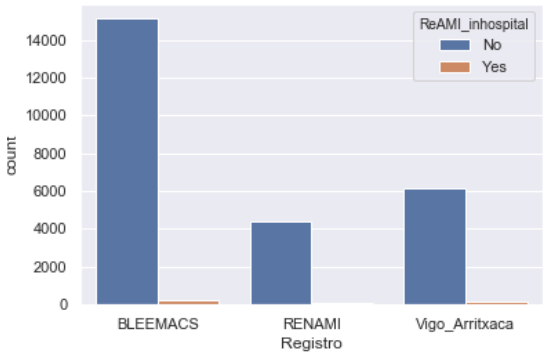

Femoral_Access
No
Yes
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

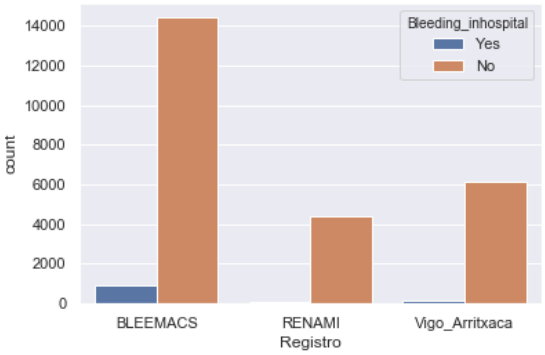

Multivessel
Yes
No
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

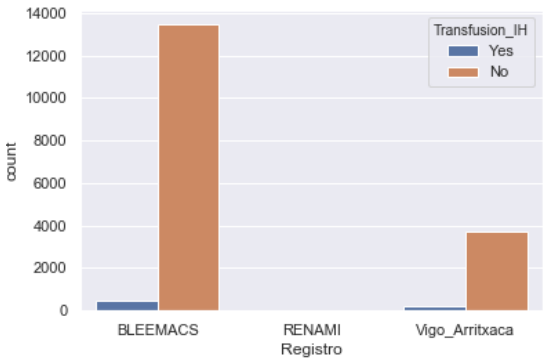

DrugElutingStent
No
Yes
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

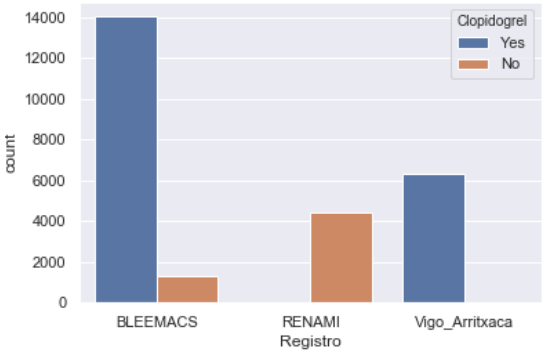

NoStent
0
1
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

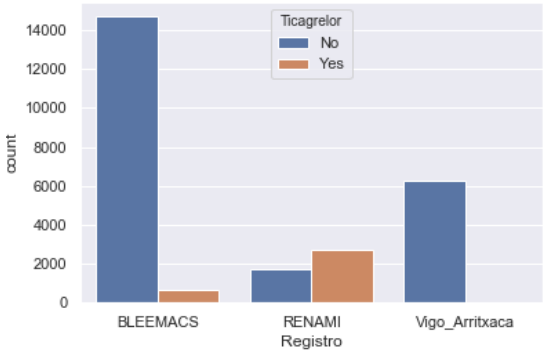

Revascularization_Complete
No
Yes
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

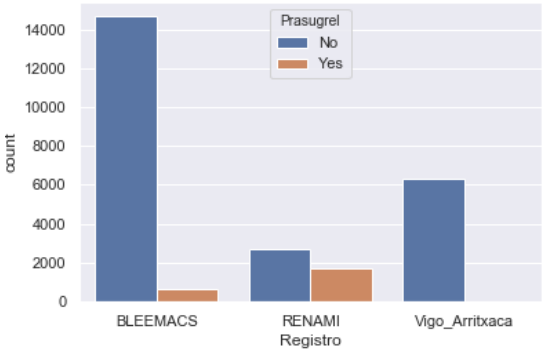

ReAMI_inhospital
No
Yes
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

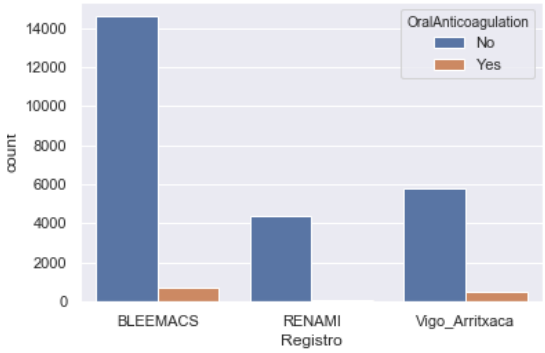

Bleeding_inhospital
Yes
No
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

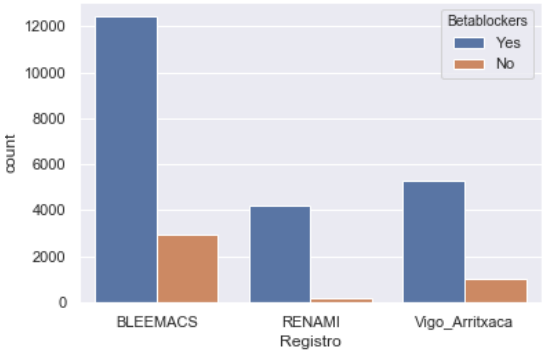

Transfusion_IH
Yes
No
count
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

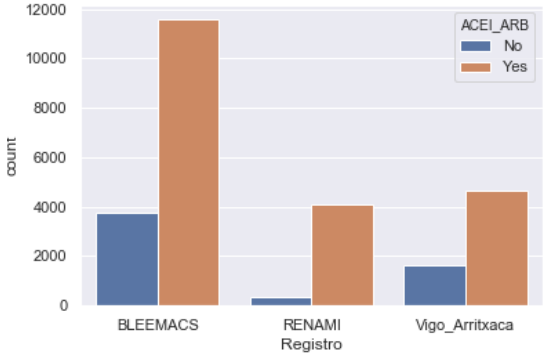

Clopidogrel
Yes
No
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

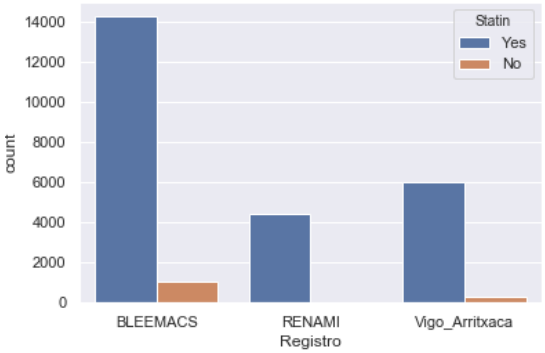

Ticagrelor
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

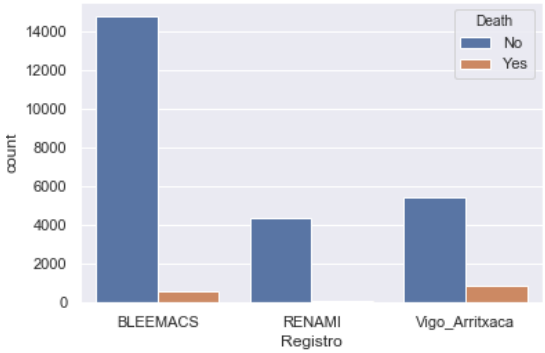

Prasugrel
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro


OralAnticoagulation
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro


Betablockers
Yes
No
count
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro


ACEI_ARB
No
Yes
count
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro


Statin
Yes
No
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro


Death
No
Yes
count
14000
12000
10000
8000
6000
4000
2000
0
BLEEMACS
RENAMI
Vigo_Arritxaca
Registro

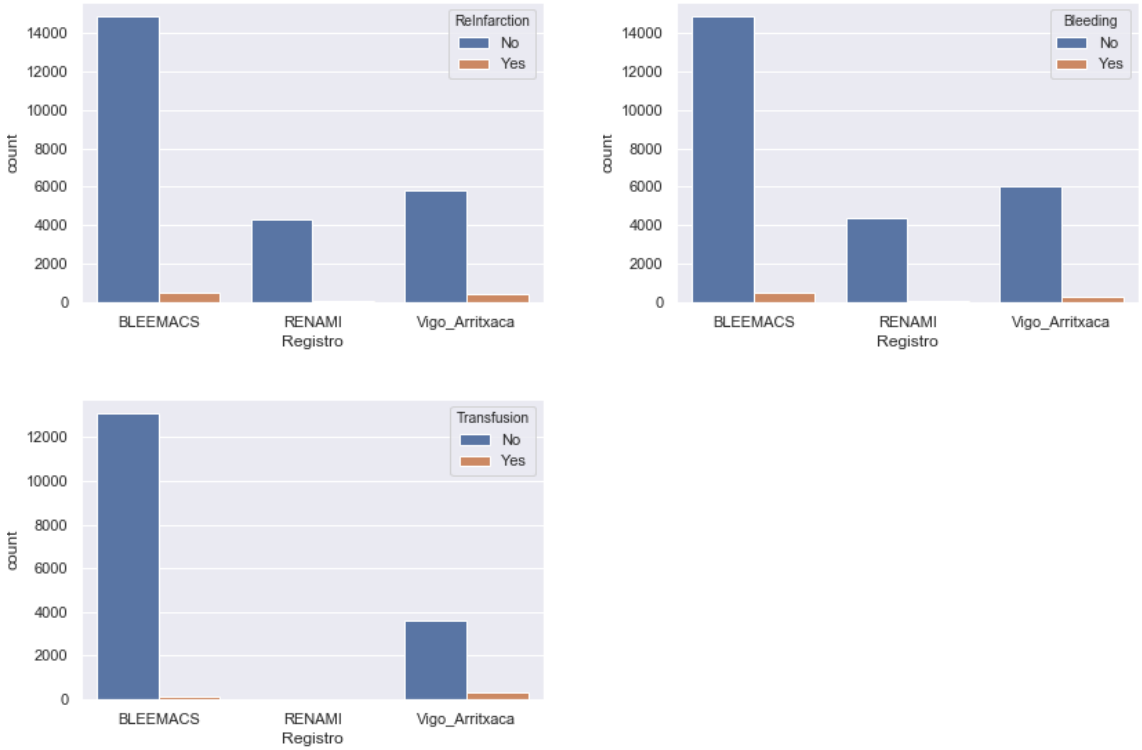


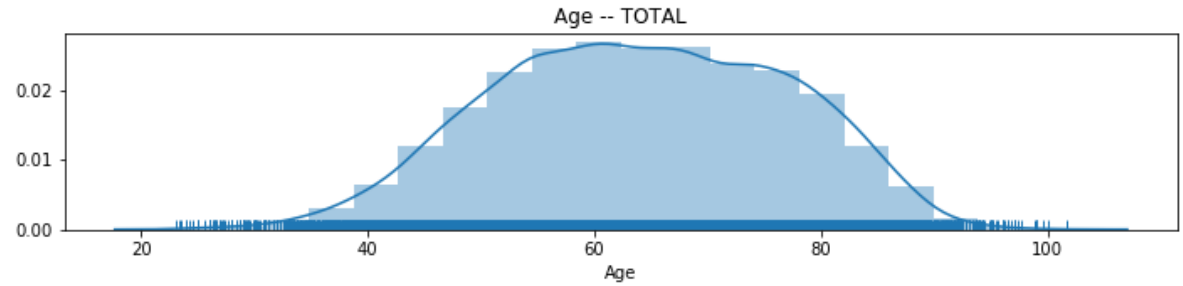
Ilustración 1. Gráficos variables categóricas por registro

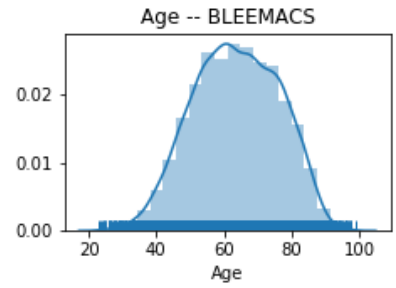
La distribución de la muestra dividida por registros en cada variable es la siguiente:

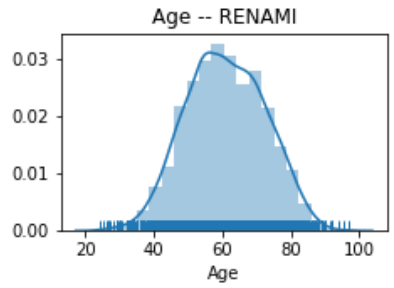
Tabla 1. Distribución de la muestra dividida por registros en cada variable.

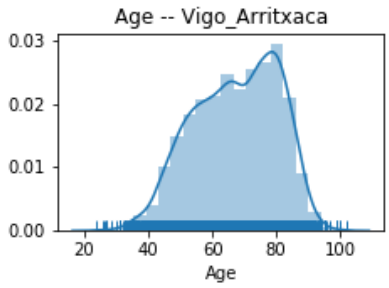
Variable	Registro		
Sexo femenino	BLEEMACS 23.27	RENAMI 20.82	VIGO_ARITXACA 25.34
Diabetes mellitus (%)	BLEEMACS 24.01	RENAMI 29.89	VIGO_ARITXACA 31.12
Hipertensión (%)	BLEEMACS 58.68	RENAMI 54.00	VIGO_ARITXACA 67.17
Dislipemia (%)	BLEEMACS 52.16	RENAMI 53.77	VIGO_ARITXACA 67.01
Enfermedad vascular periférica (%)	BLEEMACS 5.86	RENAMI 2.05	VIGO_ARITXACA 9.66
Infarto agudo de miocardio (anterior) (%)	BLEEMACS 11.94	RENAMI 16.46	VIGO_ARITXACA 32.52
Intervención percutánea coronaria (anterior) (%)	BLEEMACS 12.49	RENAMI 17.93	VIGO_ARITXACA 21.97
Baipás coronario (anterior) (%)	BLEEMACS 3.31	RENAMI 0.83	VIGO_ARITXACA 4.78
Accidente cerebrovascular (anterior) (%)	BLEEMACS 5.85	RENAMI 5.15	VIGO_ARITXACA 8.12
Cáncer (%)	BLEEMACS	RENAMI	VIGO_ARITXACA

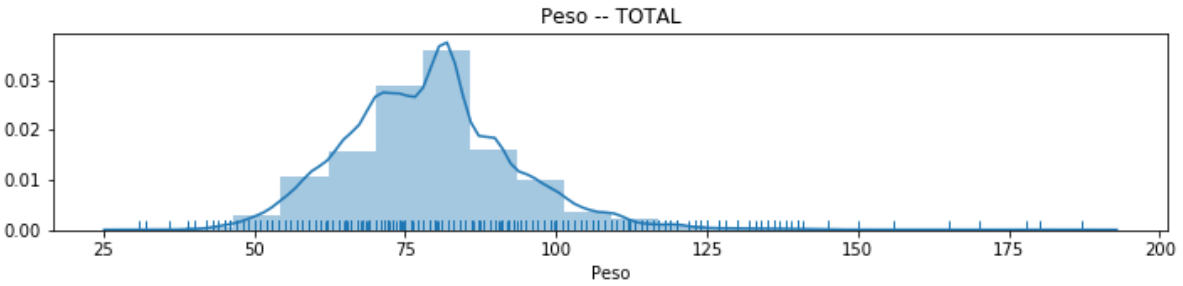
	6.01	4.50	8.55
Sangrado (anterior) (%)	BLEEMACS 5.18	RENAMI 2.41	VIGO_ARITXACA 3.19
Infarto agudo de miocardio con elevación del ST (%)	BLEEMACS 57.96	RENAMI 58.05	VIGO_ARITXACA 36.75
Angina inestable	BLEEMACS 13.38	RENAMI 9.09	VIGO_ARITXACA 14.24
Killip2orMore (%)	BLEEMACS 10.64	RENAMI 4.92	VIGO_ARITXACA 17.17
Acceso femoral (%)	BLEEMACS 52.75	RENAMI 23.69	VIGO_ARITXACA 10.09
Multivessel (%)	BLEEMACS 38.62	RENAMI 45.07	VIGO_ARITXACA 38.45
Stent farmacológico (%)	BLEEMACS 39.32	RENAMI 34.28	VIGO_ARITXACA 48.77
No stent (%)	BLEEMACS 0	RENAMI 0	VIGO_ARITXACA 18.25
Revascularización completa (%)	BLEEMACS 69.67	RENAMI 85.45	VIGO_ARITXACA 57.58
Reinfarto en el hospital (%)	BLEEMACS 1.28	RENAMI 1.28	VIGO_ARITXACA 2.63
Sangrado en el hospital (%)	BLEEMACS 5.94	RENAMI 1.17	VIGO_ARITXACA 1.93
Transfusión IH (%)	BLEEMACS 3.29		VIGO_ARITXACA 4.20
Clopidogrel (%)	BLEEMACS 91.56	RENAMI 0	VIGO_ARITXACA 100
Ticagrelor (%)	BLEEMACS 4.10	RENAMI 61.55	VIGO_ARITXACA 0
Prasugrel (%)	BLEEMACS 4.32	RENAMI 38.44	VIGO_ARITXACA 0
Anticuagulantes orales (%)	BLEEMACS 4.77	RENAMI 1.49	VIGO_ARITXACA 7.75
Betabloqueantes (%)	BLEEMACS 80.97	RENAMI 95.25	VIGO_ARITXACA 83.91
ACEI_ARB (%)	BLEEMACS 75.48	RENAMI 92.35	VIGO_ARITXACA 73.87
Estatina (%)	BLEEMACS 93.06	RENAMI 99.29	VIGO_ARITXACA 95.88
Reinfarto (%)	BLEEMACS 3.18	RENAMI 2.08	VIGO_ARITXACA 7.01
Sangrado (%)	BLEEMACS 3.20	RENAMI 1.62	VIGO_ARITXACA 4.54
Fallecimiento (%)	BLEEMACS 3.69	RENAMI 1.51	VIGO_ARITXACA 13.99
Transfusión (%)	BLEEMACS 1.19		VIGO_ARITXACA 8.58

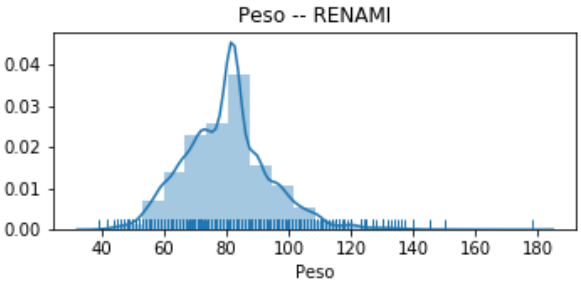

Age -- TOTAL
0.02
0.01
0.00
20
40
60
80
100
Age

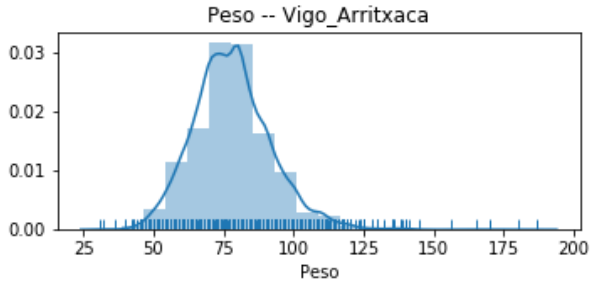

Age -- BLEEMACS
0.02
0.01
0.00
20
40
60
80
100
Age

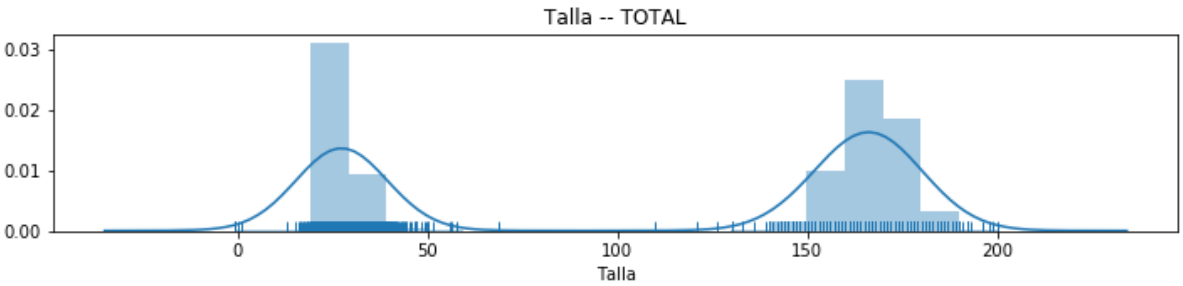

Age -- RENAMI
0.03
0.02
0.01
0.00
20
40
60
80
100
Age

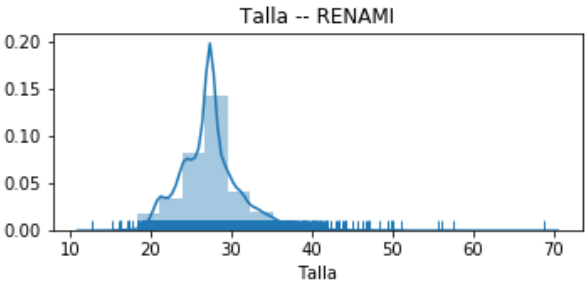

Age -- Vigo_Arritxaca
0.03
0.02
0.01
0.00
20
40
60
80
100
Age

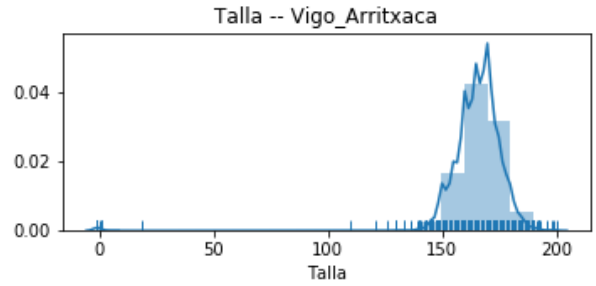

Peso -- TOTAL
0.03
0.02
0.01
0.00
25
50
75
100
125
150
175
200
Peso

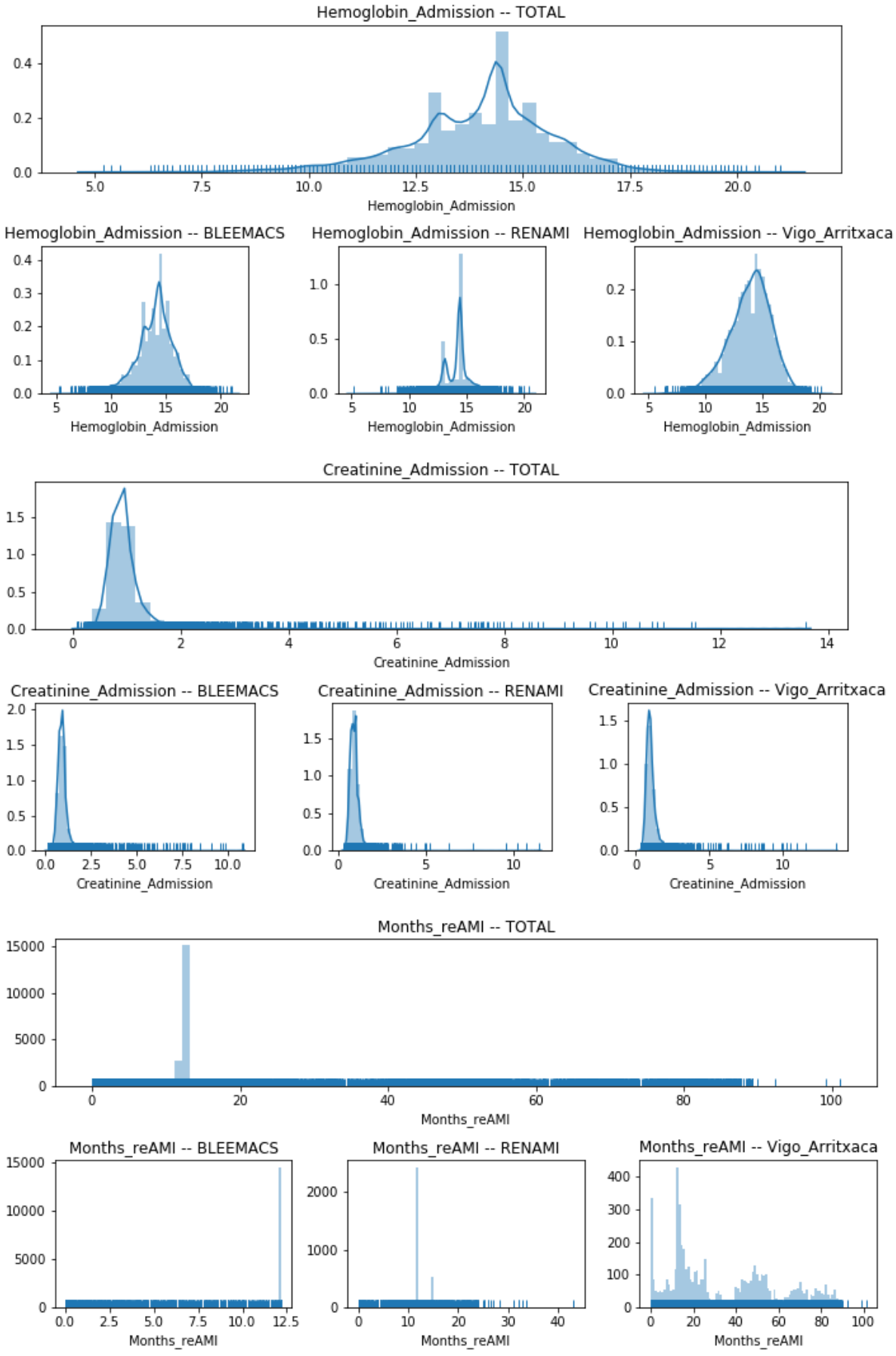

Peso -- RENAMI
0.04
0.03
0.02
0.01
0.00
40
60
80
100
120
140
160
180
Peso


Peso -- Vigo_Arritxaca
0.03
0.02
0.01
0.00
25
50
75
100
125
150
175
200
Peso


Talla -- TOTAL
0.03
0.02
0.01
0.00
0
50
100
150
200
Talla


Talla -- RENAMI
0.20
0.15
0.10
0.05
0.00
10
20
30
40
50
60
70
Talla


Talla -- Vigo_Arritxaca
0.04
0.02
0.00
0
50
100
150
200
Talla


Hemoglobin_Admission -- TOTAL
Hemoglobin_Admission
Hemoglobin_Admission -- BLEEMACS
Hemoglobin_Admission
Hemoglobin_Admission -- RENAMI
Hemoglobin_Admission
Hemoglobin_Admission -- Vigo_Arritxaca
Hemoglobin_Admission
Creatinine_Admission -- TOTAL
Creatinine_Admission
Creatinine_Admission -- BLEEMACS
Creatinine_Admission
Creatinine_Admission -- RENAMI
Creatinine_Admission
Creatinine_Admission -- Vigo_Arritxaca
Creatinine_Admission
Months_reAMI -- TOTAL
Months_reAMI
Months_reAMI -- BLEEMACS
Months_reAMI
Months_reAMI -- RENAMI
Months_reAMI
Months_reAMI -- Vigo_Arritxaca
Months_reAMI

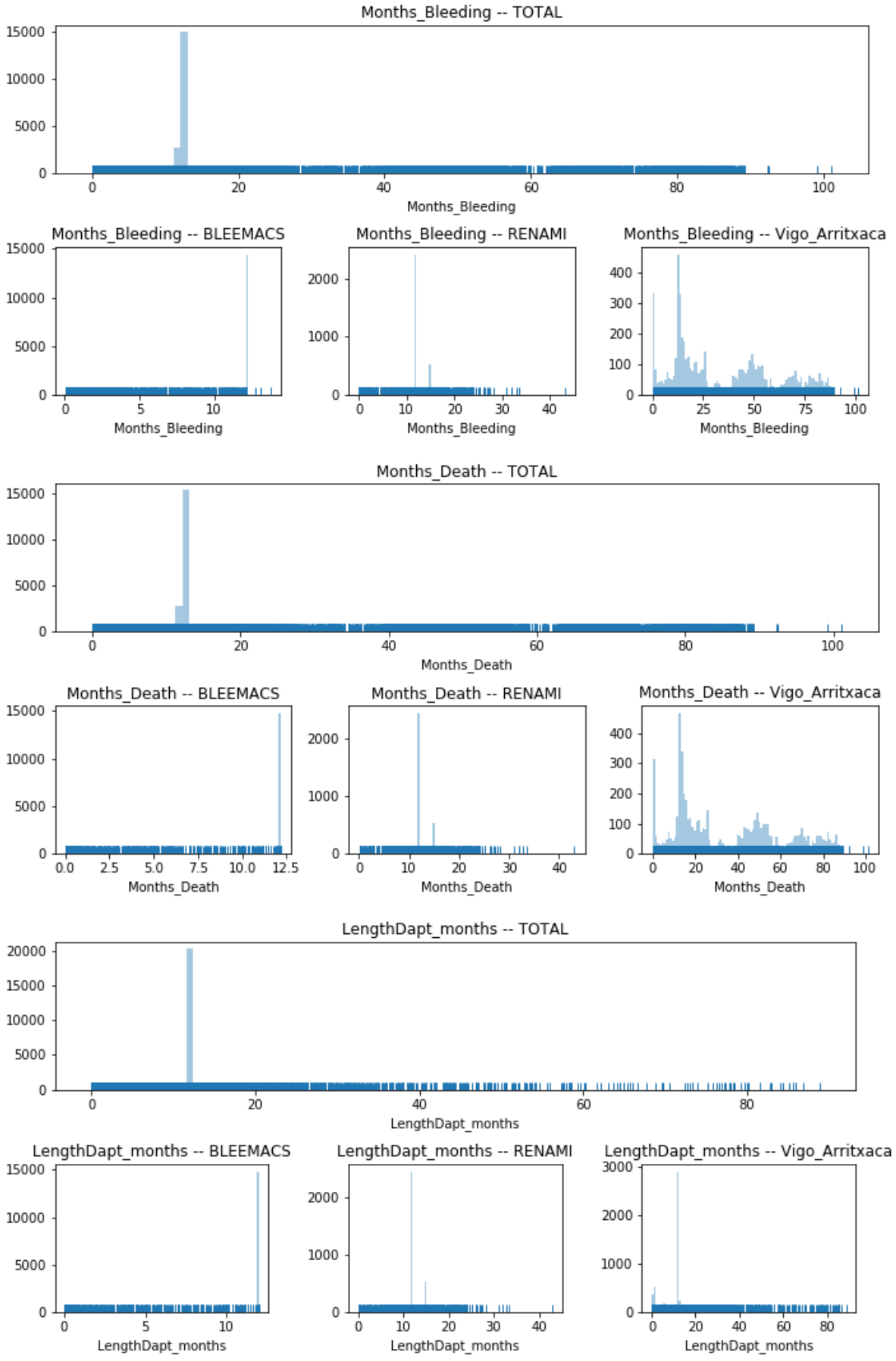


Ilustración 2. Histogramas de las variables continuas en la población total y por registro

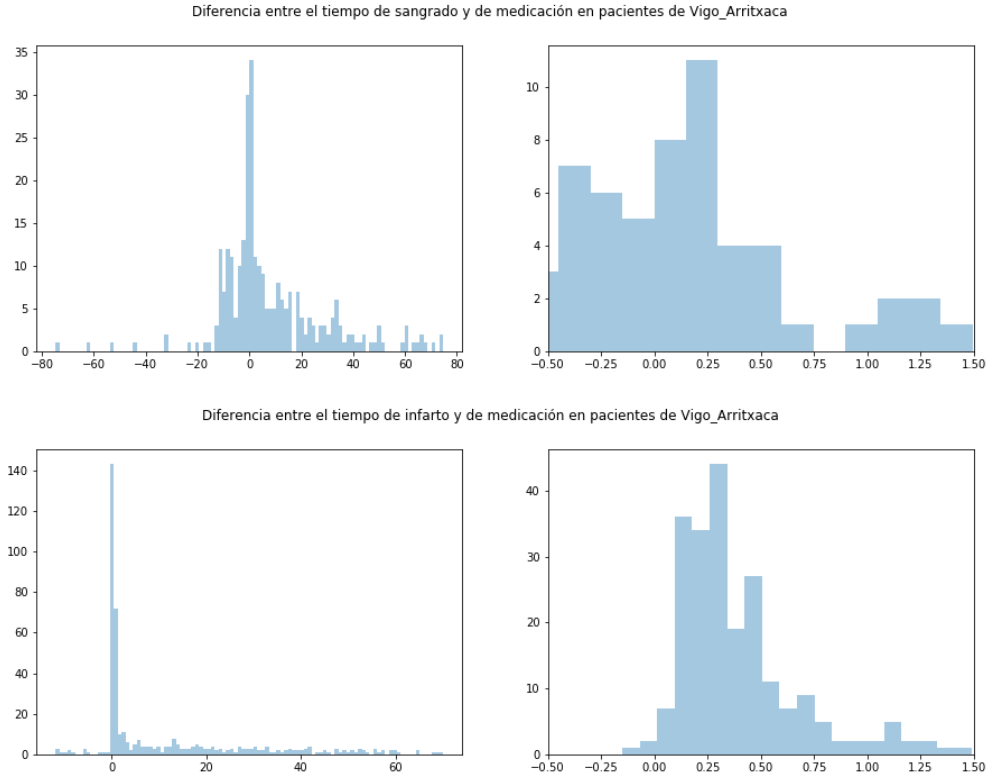


Ilustración 3. Histograma resultante de enfrentar el tiempo de evento con respecto al tiempo de medicación en el registro Vigo_Arritxaca.

Tabla 2. Resultados del ajuste de hiperparámetros.

Hemorragia															
Regresión logística															
Hiperparámetros				Resultados del entrenamiento (AUC)											
C	**l1_ratio**	**penalty**	**Solver**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
0,1	0	elasticnet	saga	0,7401	0,6995	0,7639	**0,6183**	0,5994	0,5613	0,7002	0,7590	0,4695	0,6150	0,6526	0,0912
0,1	0,5	elasticnet	saga	0,7391	0,7017	0,7683	0,6161	0,6124	0,5686	0,7018	0,7647	0,4787	0,6314	0,6583	0,0887
0,1	1	elasticnet	saga	0,7372	**0,7034**	**0,7719**	0,6153	**0,6196**	**0,5778**	**0,7039**	**0,7680**	**0,4904**	**0,6315**	**0,6619**	0,0860
1	0	elasticnet	saga	**0,7402**	0,6986	0,7635	0,6182	0,5978	0,5592	0,6996	0,7567	0,4672	0,6090	0,6510	0,0918
1	0,5	elasticnet	saga	0,7400	0,6999	0,7639	**0,6183**	0,5997	0,5600	0,6998	0,7574	0,4682	0,6108	0,6518	0,0915
1	1	elasticnet	saga	0,7400	0,7006	0,7643	0,6181	0,6010	0,5608	0,6999	0,7583	0,4693	0,6125	0,6525	0,0913
10	0	elasticnet	saga	**0,7402**	0,6985	0,7634	0,6182	0,5976	0,5589	0,6995	0,7565	0,4670	0,6080	0,6508	0,0919
10	0,5	elasticnet	saga	0,7401	0,6986	0,7634	0,6182	0,5978	0,5590	0,6996	0,7565	0,4671	0,6083	0,6509	0,0918
10	1	elasticnet	saga	**0,7402**	0,6987	0,7635	0,6182	0,5979	0,5592	0,6996	0,7566	0,4673	0,6085	0,6510	0,0918
Support Vector Machine															
Hiperparámetros				Resultados del entrenamiento (AUC)											
C		**Kernel**		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
0,1		linear		**0,5416**	**0,4498**	0,3594	**0,5493**	0,5041	**0,3892**	0,4645	**0,4113**	0,4568	0,4775	0,4603	0,0588
1		linear		0,4137	0,3337	0,3188	0,4755	0,4100	0,3370	0,5516	0,3811	**0,7024**	0,3781	0,4302	0,1127
10		linear		0,4454	0,4026	**0,4662**	0,3142	**0,5648**	0,3338	**0,6135**	0,3595	0,5925	**0,5445**	**0,4637**	0,1049
Random forest															
Hiperparámetros				Resultados del entrenamiento (AUC)											
max_depth		**max_features**		**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
1		auto		0,7060	0,7034	0,7541	0,6249	0,6405	**0,6448**	0,7031	0,8112	0,6758	0,6725	0,6936	0,0534
1		1		0,7158	0,7018	0,7420	0,6077	0,6254	0,5606	0,6942	**0,8444**	0,6416	0,6327	0,6766	0,0767
2		auto		0,7130	**0,7179**	**0,7618**	0,6171	0,6398	0,5923	**0,7118**	0,8222	**0,6985**	0,6808	**0,6955**	0,0645
2		1		**0,7194**	0,7169	0,7496	**0,6251**	0,6024	0,5286	0,7083	0,8198	0,6024	0,6794	0,6752	0,0812
5		auto		0,7191	0,7128	0,7554	0,6087	0,6639	0,6351	0,6888	0,8171	0,6408	0,6919	0,6934	0,0587
5		1		0,7059	0,6979	0,7403	0,5891	0,6438	0,5666	0,7039	0,7574	0,5640	0,6755	0,6644	0,0668
		auto		0,6377	0,6142	0,6841	0,5976	**0,7521**	0,6079	0,5574	0,6827	0,5164	**0,7665**	0,6417	0,0761
		1		0,6295	0,6323	0,6244	0,5943	0,7484	0,5683	0,5829	0,6264	0,5455	0,7422	0,6294	0,0641

Reinfarto

Regresión logística

Hiperparámetros				Resultados del entrenamiento (AUC)											
C	**l1_ratio**	**penalty**	**Solver**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
0,1	0	elasticnet	saga	0,6187	0,6662	0,6394	0,5806	**0,7226**	**0,8912**	0,7415	0,6808	0,6011	0,6553	0,6797	0,0851
0,1	0,5	elasticnet	saga	0,6208	0,6669	0,6447	0,5840	0,7214	0,8863	0,7410	0,6847	**0,6035**	0,6590	0,6812	0,0827
0,1	1	elasticnet	saga	**0,6217**	**0,6670**	**0,6489**	**0,5858**	0,7171	0,8812	0,7408	**0,6893**	0,6000	**0,6612**	**0,6813**	0,0811
1	0	elasticnet	saga	0,6171	0,6668	0,6369	0,5784	0,7220	0,8909	**0,7416**	0,6807	0,5986	0,6550	0,6788	0,0858
1	0,5	elasticnet	saga	0,6174	0,6669	0,6384	0,5787	0,7218	0,8907	0,7414	0,6811	0,5987	0,6555	0,6791	0,0856
1	1	elasticnet	saga	0,6178	0,6668	0,6400	0,5791	0,7218	0,8902	0,7414	0,6815	0,5991	0,6555	0,6793	0,0852
10	0	elasticnet	saga	0,6170	0,6668	0,6365	0,5781	0,7220	0,8909	0,7415	0,6806	0,5983	0,6550	0,6787	0,0859
10	0,5	elasticnet	saga	0,6170	0,6668	0,6366	0,5781	0,7220	0,8909	0,7415	0,6806	0,5983	0,6551	0,6787	0,0858
10	1	elasticnet	saga	0,6170	0,6668	0,6369	0,5781	0,7219	0,8909	0,7415	0,6807	0,5984	0,6551	0,6787	0,0858

Support Vector Machine

Hiperparámetros		Resultados del entrenamiento (AUC)											
C	**Kernel**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
0,1	linear	0,4730	0,5223	**0,6061**	**0,6797**	**0,4061**	0,0929	**0,4903**	0,4699	**0,5740**	0,4505	0,4765	0,1492
1	linear	**0,5302**	**0,5989**	0,5223	0,5932	0,3976	0,2802	0,3960	0,5434	0,4592	0,5658	0,4887	0,0980
10	linear	0,4415	0,5353	0,5211	0,5348	0,3531	**0,7977**	0,3987	**0,5674**	0,5085	**0,6380**	**0,5296**	0,1190

Random forest

Hiperparámetros		Resultados del entrenamiento (AUC)											
max_depth	**max_features**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**media**	**desviación**
1	auto	0,6617	0,6339	0,6964	0,6606	0,6109	0,8301	0,6511	0,6319	0,5882	0,6331	0,6598	0,0633
1	1	**0,6872**	0,6218	0,6980	**0,6681**	0,6969	0,8487	0,6514	0,7213	0,6836	0,6285	0,6906	0,0606
2	auto	0,6410	0,6195	0,7258	0,6321	0,6241	0,8439	0,6607	0,6566	0,6645	0,6350	0,6703	0,0647
2	1	0,6615	0,6376	**0,7600**	0,6475	0,6751	0,8811	**0,7267**	**0,7409**	**0,7119**	0,6422	**0,7084**	0,0709
5	auto	0,6468	**0,6420**	0,6912	0,5758	0,6607	0,8610	0,6785	0,6725	0,6246	0,6254	0,6679	0,0716
5	1	0,6584	0,6375	0,6861	0,6498	0,7134	**0,9039**	0,7155	0,7009	0,6694	**0,6538**	0,6989	0,0731
	auto	0,5875	0,5929	0,6432	0,5228	0,6949	0,7989	0,6595	0,6069	0,6199	0,5728	0,6299	0,0724
	1	0,6201	0,6137	0,6114	0,5304	**0,7396**	0,7841	0,7198	0,6117	0,6139	0,4677	0,6312	0,0902

3. Código

0_ExploreDatabase

1. **Descriptive of the variables**
    a. Categorical variable:
        - Percentages.
        - Bar plots.
    b. Continuos variable:
        - Histogram.
1. **Time**: total times on variables: Months_Death, Months_reAMI, Months_Bleeding y LengthDapt_months.
    a. Absolute time
    b. Difference from medication time
1. **Survival Analysis**
1. **Frequency table**

In [ ]:

```
# Import librarys

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import seaborn as sns
from datetime import date, datetime
from sksurv.nonparametric import kaplan_meier_estimator

pd.options.display.max_columns = None

%pylab inline
```

In [ ]:

```
# Import data

df = pd.read_excel('../../data/PACS_DAPT_completo.xlsx')
```

1. Descriptive of the variables¶

In [ ]:

```
df.head()
```

In [ ]:

```
print("Variables on BleeMACS registry: ", df.loc[(df['Registro']=='BleeMACS'),].shape[0],
    "/nVariables on RENAMI registry: ", df.loc[(df['Registro']=='RENAMI'),].shape[0],
    "/nVariables on Vigo_Arritxaca registry: ", df.loc[(df['Registro']=='Vigo_Arritxaca'),].shape[0])
```

In [ ]:

df.columns

A. Categorical variable¶

In [ ]:

```
var_cat = ['Sex', 'Diabetes', 'Hypertension', 'Dyslipemia', 'PeripheralArteryDisease', 'PriorMyocardialInfarction',
        'PriorPCI','PriorCABG', 'PriorStroke', 'Cancer', 'Prior_Bleeding', 'STEMI', 'UnstableAngina',
        'Killip2orMore', 'LVEFless40', 'LVEFgroups', 'Femoral_Access', 'Multivessel', 'DrugElutingStent',
        'NoStent','Revascularization_Complete', 'ReAMI_inhospital', 'Bleeding_inhospital', 'Transfusion_IH',
        'Clopidogrel', 'Ticagrelor', 'Prasugrel', 'OralAnticoagulation', 'Betablockers', 'ACEI_ARB', 'Statin',
        'Death', 'ReInfarction', 'Bleeding', 'Transfusion']
```

- Percentaje¶

In [ ]:

```
for el in var_cat:
    print(df[el].value_counts(normalize=True) * 100,
          '\n-----------------------------------------------------------\n')
```

In [ ]:

```
for el in var_cat:
    print(df.groupby('Registro')[el].value_counts(normalize=True) * 100,
          '\n-----------------------------------------------------------\n')
```

- Bar plots¶

In [ ]:

```
hue = "Sex"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Diabetes"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Hypertension"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Dyslipemia"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "PeripheralArteryDisease"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "PriorMyocardialInfarction"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "PriorPCI"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "PriorCABG"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "PriorStroke"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Cancer"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Prior_Bleeding"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "STEMI"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "UnstableAngina"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Killip2orMore"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "LVEFless40"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "LVEFgroups"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Femoral_Access"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Multivessel"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "DrugElutingStent"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "NoStent"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Revascularization_Complete"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "ReAMI_inhospital"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Bleeding_inhospital"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Transfusion_IH"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Clopidogrel"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Ticagrelor"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Prasugrel"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "OralAnticoagulation"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Betablockers"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "ACEI_ARB"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Statin"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Death"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "ReInfarction"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Bleeding"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

In [ ]:

```
hue = "Transfusion"
sns_plot = sns.countplot(x="Registro", hue=hue,data=df)
# sns_plot.figure.savefig(str('BarPlot_' + hue +'.png'))
```

B. Continuos variable¶

In [ ]:

```
var_cont = ['Year', 'Age', 'Peso', 'Talla', 'Hemoglobin_Admission', 'Creatinine_Admission', 'Months_D
eath',
            'Months_reAMI', 'Months_Bleeding', 'LengthDapt_months']
```

- Histograms¶

In [ ]:

```
def histograma(variable, n_bins, registro = ['BLEEMACS', 'RENAMI', 'Vigo_Arritxaca']):
    fig = plt.figure(tight_layout=True, figsize= (10,5))
    gs = gridspec.GridSpec(2, len(registro))
    i =  0
    ax = fig.add_subplot(gs[0,:])
    sns.distplot(df[variable], bins=n_bins, rug=True, kde=False, ax= ax)
    ax.set_title(variable +' -- TOTAL')
    # ax.set_ylabel('YLabel0')
    # ax.set_xlabel('XLabel0')
    for reg in registro:
        ax = fig.add_subplot(gs[1, i])
        sns.distplot(df.loc[df['Registro'] == str(reg), variable], bins=n_bins, rug=True, kde=False, ax= ax)
        ax.set_title(variable +' -- ' +reg)
    #     ax.set_ylabel('YLabel1 %d' % i)
    #     ax.set_xlabel('XLabel1 %d' % i)
        i = i+1
        if i == 0:
            for tick in ax.get_xticklabels():
                tick.set_rotation(55)
    fig.align_labels()  # same as fig.align_xlabels(); fig.align_ylabels()
#     fig.savefig(str('HistPlot_' + variable +'.png'))
```

In [ ]:

```
histograma('Age',20)
```

In [ ]:

```
histograma('Peso', 20, ['RENAMI', 'Vigo_Arritxaca'])
```

In [ ]:

```
histograma('Talla', 20, ['RENAMI', 'Vigo_Arritxaca'])
```

In [ ]:

```
histograma('Hemoglobin_Admission', 50)
```

In [ ]:

```
histograma('Creatinine_Admission', 50)
```

In [ ]:

```
histograma('Months_Death', 100)
```

In [ ]:

```
histograma('Months_reAMI', 100)
```

In [ ]:

```
histograma('Months_Bleeding', 100)
```

In [ ]:

```
histograma('LengthDapt_months', 100)
```

---

2. Time¶

Total times on variables: Months_Death, Months_reAMI, Months_Bleeding y LengthDapt_months.¶

A. Absolute time¶

In [ ]:

```
def tiempos(variable, n_bins1=100, n_bins2=500, xlim = [9,15]):
    fig, (ax1, ax2) = plt.subplots(1 ,2, figsize= (15,5))
    fig.suptitle(variable.name)
    sns.distplot(variable, bins=n_bins1, rug=False, kde=False, ax= ax1)
#     ax1.hist(variable, bins=n_bins1)
    sns.distplot(variable, bins=n_bins2, rug=False, kde=False, ax= ax2)
#     ax2.hist(variable, bins=n_bins2)
    ax2.set_xlim(xlim)
```

In [ ]:

```
tiempos(df['Months_Death'])
```

In [ ]:

```
print('Months_Death con muerte = SI')
tiempos(df.loc[df['Death'] == 'Yes', 'Months_Death'], xlim =[0,12])
```

In [ ]:

```
tiempos(df['Months_reAMI'])
```

In [ ]:

```
print('Months_reAMI con ReInfarction = SI')
tiempos(df.loc[df['ReInfarction'] == 'Yes', 'Months_reAMI'], xlim =[0,15])
```

In [ ]:

```
tiempos(df['Months_Bleeding'])
```

In [ ]:

```
print('Months_Bleeding con Bleeding = SI')
tiempos(df.loc[df['Bleeding'] == 'Yes', 'Months_Bleeding'], xlim =[0,15])
```

In [ ]:

```
tiempos(df['LengthDapt_months'])
```

B. Difference from medication time¶

In [ ]:

```
def diferencia(save,title, variable, n_bins1=100, n_bins2=1000, xlim = [-0.5,1.5]):
    fig, (ax1, ax2) = plt.subplots(1 ,2, figsize= (15,5))
    fig.suptitle(title)
    sns.distplot(variable, bins=n_bins1, rug=False, kde=False, ax= ax1)
    sns.distplot(variable, bins=n_bins2, rug=False, kde=False, ax= ax2)
    ax2.set_xlim(xlim)
#     fig.savefig(str('HistPlot_' + save +'.png'))
```

In [ ]:

```
diferencia('difREAMI',
    'Diferencia entre el tiempo de infarto y de medicación en pacientes de Vigo_Arritxaca',
    df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['ReInfarction'] == 'Yes'), 'Months_reAMI'] - df.loc[(
df['Registro']=='Vigo_Arritxaca') & (df['ReInfarction'] == 'Yes'), 'LengthDapt_months'])
```

In [ ]:

```
diferencia('DifBLEEDING',
    'Diferencia entre el tiempo de sangrado y de medicación en pacientes de Vigo_Arritxaca',
    df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Bleeding'] == 'Yes'), 'Months_Bleeding'] - df.loc[(d
f['Registro']=='Vigo_Arritxaca') & (df['Bleeding'] == 'Yes'), 'LengthDapt_months'])
```

---

3. Survival Analysis¶

In [ ]:

```
#variables que es necesario realizar el cambio

replace_vars = ['ReInfarction','Bleeding', 'Transfusion', 'Death']

#creamos un diccionario con los cambios
dic = {'Yes': np.True_, 'No': np.False_, np.nan:np.nan,'NaN':np.nan, 1:1, 0:0}

#Realizamos el cambio
for i in replace_vars:
    #print(i)
    df[i] = df[i].apply(lambda x:dic[x])
```

In [ ]:

```
time, survival_prob = kaplan_meier_estimator(df["ReInfarction"], df["Months_reAMI"])
plt.step(time, survival_prob, where="post", label = 'Reinfarto')
# plt.xlim([0,12])
plt.ylabel("est. probability of survival $\hat{S}(t)$")
plt.xlabel("time $t$")
```

In [ ]:

```
df.loc[(df.Months_reAMI >=90),['Registro','ReInfarction','Months_reAMI']]
```

In [ ]:

```
time, survival_prob = kaplan_meier_estimator(df.loc[df.Registro == "Vigo_Arritxaca","Bleeding"], df
.loc[df.Registro == "Vigo_Arritxaca","Months_Bleeding"])
plt.step(time, survival_prob, where="post")
plt.ylim([.65,1])
plt.ylabel("est. probability of survival $\hat{S}(t)$")
plt.xlabel("time $t$")
```

In [ ]:

```
df.loc[(df.Bleeding == True)&(df.Months_Bleeding >=70),['Registro','Bleeding','Months_Bleeding']]
```

In [ ]:

```
time, survival_prob = kaplan_meier_estimator(df["Death"], df["Months_Death"])
plt.step(time, survival_prob, where="post", label = 'Fallecimiento')
plt.ylim([.65,1])
plt.ylabel("est. probability of survival $\hat{S}(t)$")
plt.xlabel("time $t$")
```

In [ ]:

```
df.loc[(df.Bleeding == True)&(df.Months_Bleeding >=70),['Registro','Death','Months_Death']]
```

In [ ]:

```
time, survival_prob = kaplan_meier_estimator(df["ReInfarction"], df["Months_reAMI"])
plt.step(time, survival_prob, where="post", label = 'Reinfarto')
time, survival_prob = kaplan_meier_estimator(df["Bleeding"], df["Months_Bleeding"])
plt.step(time, survival_prob, where="post", label = 'Hemorragia')
time, survival_prob = kaplan_meier_estimator(df["Death"], df["Months_Death"])
plt.step(time, survival_prob, where="post", label = 'Fallecimiento')
# plt.xlim([0,24])
plt.ylabel("est. probability of survival $\hat{S}(t)$")
plt.xlabel("time $t$")

plt.legend(loc="lower left", fontsize = 10)
# plt.savefig('supervivencia.png')
```

---

4. Frequency tables¶

This part will be done after pre-processing the data.

In [ ]:

```
# Import data

df_pre = pd.read_csv('../../data/PACS_DAPT_preprocess.csv')
df_pre = df_pre.drop('Unnamed: 0', axis=1)
```

In [ ]:

```
Train = df_pre.loc[(df_pre.Registro == 'BLEEMACS')|(df_pre.Registro == 'RENAMI'), ].copy()
Test = df_pre.loc[(df_pre.Registro == 'Vigo_Arritxaca'), ].copy()
```

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', Train.loc[(Train['reAMIless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', Train.loc[(Train['reAMIless12Dapt'] == 0) , 'Id'].count(),
    '\n-----------------------------------------------------',
    '\nConjunto de validación:',
    '\n- Sí:', Test.loc[(Test['reAMIless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', Test.loc[(Test['reAMIless12Dapt'] == 0) , 'Id'].count())
```

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n', Train['reAMIless12Dapt'].value_counts(normalize=True) * 100,
    '\n-----------------------------------------------------',
    '\nConjunto de validación:',
    '\n', Test['reAMIless12Dapt'].value_counts(normalize=True) * 100)
```

In [ ]:

```
print('Pacientes con hemorragia antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', Train.loc[(Train['bleedingless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', Train.loc[(Train['bleedingless12Dapt'] == 0) , 'Id'].count(),
    '\n-----------------------------------------------------',
    '\nConjunto de validación:',
    '\n- Sí:', Test.loc[(Test['bleedingless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', Test.loc[(Test['bleedingless12Dapt'] == 0) , 'Id'].count())
```

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n', Train['bleedingless12Dapt'].value_counts(normalize=True) * 100,
    '\n-----------------------------------------------------',
    '\nConjunto de validación:',
    '\n', Test['bleedingless12Dapt'].value_counts(normalize=True) * 100)
```

Table on the number of reinfarctions, bleedings and deaths.

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', df_pre.loc[(df_pre['reAMIless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', df_pre.loc[(df_pre['reAMIless12Dapt'] == 0) , 'Id'].count(),
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n', df_pre['reAMIless12Dapt'].value_counts(normalize=True) * 100,
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con hemorragia  antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', df_pre.loc[(df_pre['bleedingless12Dapt'] == 1) , 'Id'].count(),
    '\n- No:', df_pre.loc[(df_pre['bleedingless12Dapt'] == 0) , 'Id'].count(),
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con ReAMI antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n', df_pre['bleedingless12Dapt'].value_counts(normalize=True) * 100,
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con muerte antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', df_pre.loc[(df_pre['Death'] == 1) & (df_pre['Months_Death'] <= 12), 'Id'].count(),
    '\n- Porcentaje: ', df_pre.loc[(df_pre['Death'] == 1) & (df_pre['Months_Death'] <= 12), 'Id'].count()
* 100 / 26076,
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con muerte Hemorragia antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', df_pre.loc[(df_pre['Death'] == 1) & (df_pre['Months_Death'] <= 12) & (df_pre['bleedingle
ss12Dapt'] == 1), 'Id'].count(),
    '\n- Porcentaje: ', df_pre.loc[(df['Death'] == 1) & (df_pre['Months_Death'] <= 12) & (df_pre['bleedi
ngless12Dapt'] == 1), 'Id'].count()*100/633,
    '\n-----------------------------------------------------')
```

In [ ]:

```
print('Pacientes con muerte Hemorragia antes de los 12 meses',
    '\n-----------------------------------------------------',
    '\nConjunto de entrenamiento:',
    '\n- Sí:', df_pre.loc[(df_pre['Death'] == 1) & (df_pre['Months_Death'] <= 12) & (df_pre['reAMIless
```

```
12Dapt'] == 1), 'Id'].count(),
    '\n- Porcentaje: ', df_pre.loc[(df_pre['Death'] == 1) & (df_pre['Months_Death'] <= 12) & (df_pre['re
AMIless12Dapt'] == 1), 'Id'].count() * 100 / 683,
    '\n----------------------------------------------------------')
```

1_PreprocessDatabase

Import libarys

In [ ]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

%pylab inline
```

Import data

In [ ]:

```
df = pd.read_excel('../../data/PACS_DAPT_completo.xlsx')
```

In [ ]:

```
df.head()
```

Preprocess Database

2. We pass on the categorical variables to a binary classification.
3. Create a new variable "IMC".
4. Create two new variables from LVEFgroups.
5. Set the bleeding and reinflation times well in the "Vigo_Arritxaca" register.
6. Create new variables that classified infarction/bleeding at 6 months, 12 months and 24 months.

---

1. We pass on the categorical variables to a binary classification.

"Female" = 1 / "Male" = 2
"No" = 0 / "Yes" = 1

In [ ]:

```
# Variables that need to be changed

replace_vars = ['Sex','Diabetes', 'Hypertension', 'Dyslipemia', 'PeripheralArteryDisease', 'PriorMyocardialInfarction',
            'PriorPCI', 'PriorCABG', 'PriorStroke', 'Cancer', 'Prior_Bleeding', 'STEMI', 'UnstableAngina',
            'Killip2orMore', 'Femoral_Access', 'Multivessel', 'DrugElutingStent', 'Revascularization_Complete',
            'ReAMI_inhospital', 'Bleeding_inhospital', 'Transfusion_IH', 'Clopidogrel', 'Ticagrelor', 'Prasugrel',
            'OralAnticoagulation', 'Betablockers', 'ACEI_ARB', 'Statin', 'Death', 'ReInfarction','Bleeding', 'Transfusion']

# create a dictionary with the changes
dic = {'Female': 1, 'Male': 2, 'Yes': 1, 'No': 0, np.nan:np.nan,'NaN':np.nan, 1:1, 0:0}
```

In [ ]:

```
# Do the change
for i in replace_vars:
    #print(i)
    df[i] = df[i].apply(lambda x:dic[x])
```

---

2. Create a new variable "IMC"

$$IMC = \frac{peso(Kg)}{altura^2(m)}$$

- Bleemacs: No "talla" no "peso".
- Renami: Talla = IMC.
- Vigo_Arritxaca: Peso y Talla, calculate IMC

Before making changes, have several patients with inconsistent data in the database.

In [ ]:

```
df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']<100), ['Id','Talla']].shape
```

We have 25 patients where "Talla" is less than 100 cm:

- 20 patients with Talla: -1
- 3 patients with Talla: 1
- 1 patients with Talla: 0
- 1 patients with Talla: 19

Replace them with the average?

In [ ]:

```
# calculate IMC:
# - BLEEMACS haven't talla and peso

# - RENAMI the talla is equal to IMC
df['IMC'] = df.loc[df['Registro']=='RENAMI', 'Talla']

# - Vigo_Arritxaca calculate it without putting in values less than 100cm
df['IMC'] = df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']>100), 'Peso'] / ((df.loc[(df['Registr
o']=='Vigo_Arritxaca') & (df['Talla']>100), 'Talla']/100) * (df.loc[(df['Registro']=='Vigo_Arritxaca') &
(df['Talla']>100), 'Talla']/100))
```

In [ ]:

```
# The 24 patients with "Talla" = (-1, 0, 1), are replaced by the average.
IMC_media_Vigo = df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']>100), 'IMC'].mean()

df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']==-1) ,'IMC'] = IMC_media_Vigo
df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']==0) ,'IMC'] = IMC_media_Vigo
df.loc[(df['Registro']=='Vigo_Arritxaca') & (df['Talla']==1) ,'IMC'] = IMC_media_Vigo
df.loc[df['Id']==24445,'IMC'] = IMC_media_Vigo

# # El paciente con Id: 24445 tiene Talla = 19, la sustituimos por Talla = 190
# df.loc[df['Id']==24445,'Talla'] = 190
```

---

3. Create two new variables from LVEFgroups.

From "LVEFgroups" we get the variables: "LVEFless40", "LVEFless50" and "LVEFafter50".

In [ ]:

```
# Create two new columns equivalent to LVEFless40
df['LVEFless50'] = 0
df['LVEFafter50'] = 0

df.loc[~(df['LVEFgroups']=='LVEF 40-49'),'LVEFless50'] = 1

# df.loc[~(df['LVEFgroups']=='LVEF > 50'),'LVEFless50'] = 1 # Si queremos hacerlo de manera escal
onada
df.loc[~(df['LVEFgroups']=='LVEF > 50'),'LVEFafter50'] = 1
```

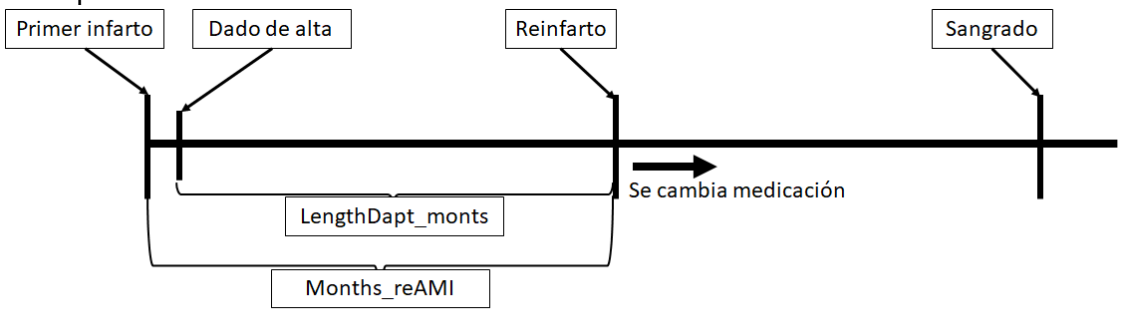
---

4. Set the bleeding and reinflation times well in the "Vigo_Arritxaca" register.

I'll use "event" to refer to reinfarction or bleeding
LengthDapt_months ": medication time, time from discharge to the next event.
Months_reAMI ": reinfarction time, time from the first heart attack to the second.
Months_Bleeding ": time of bleeding, time from first infarct to bleeding.

After any event the medication is changed.

Example:



After reinfarction, the medication is changed, so the bleeding is no longer valid.
We must set the time of measurement equal to the time of reinfarction.

create two new variables: "ReInfarctionDapt", "ReInfarctionAfterDapt", "BleedingDapt" and "BleedinfAfterDapt" in order to be able to perform a study during medication and after medication

It will be necessary to change the functions created below and make new ones.

In [ ]:

```
# Create a function to establish the medication times in Vigo_Arritxaca

def tiempoDapt(x):
    if x['Registro']=='Vigo_Arritxaca':
        x['LengthDapt_months'] = 0.5 + x['LengthDapt_months']
    return x['LengthDapt_months']

df['LengthDapt_months'] = df.apply(tiempoDapt, axis = 1)
```

In [ ]:

```
# Create different functions and then clean up the basis for more than one event.

# With this function we change the status of whether or not the patient has bled
def sangradoFalso(x):
    if x['Bleeding']==1 and x['ReInfarction'] == 1 and x['Months_reAMI'] < x['Months_Bleeding']:
        x['Bleeding'] = 0
    return x['Bleeding']

# With this function we change the state of whether or not the patient has suffered another heart attack
def reamiFalso(x):
    if x['Bleeding']==1 and x['ReInfarction'] == 1 and x['Months_Bleeding'] < x['Months_reAMI']:
        x['ReInfarction'] = 0
    return x['ReInfarction']
```

In [ ]:

```
%%time
# Run the entire functions in our database

df['Bleeding'] = df.apply(sangradoFalso, axis = 1)
df['ReInfarction']= df.apply(reamiFalso, axis = 1)
```

In [ ]:

```
# Create different functions for the creation of new variables of the events during and after medication

# With this function we see if the patient has bled during the medication
def sangradoDapt(x):
    if x['Bleeding']==1 and x['LengthDapt_months'] >= x['Months_Bleeding']:
        x['BleedingDapt'] = 1
    return x['BleedingDapt']
# With this function we see if the patient has bled after medication
def sangradoAfterDapt(x):
    if x['Bleeding']==1 and x['LengthDapt_months'] < x['Months_Bleeding']:
        x['BleedinfAfterDapt'] = 1
    return x['BleedinfAfterDapt']


# With this function we see if the patient has suffered another heart attack during medication
def reamiDapt(x):
    if x['ReInfarction'] == 1 and x['LengthDapt_months'] >= x['Months_reAMI']:
        x['ReInfarctionDapt'] = 1
    return x['ReInfarctionDapt']
```

```
# With this function we see if the patient has suffered another heart attack after medication
def reamiAfterDapt(x):
    if x['ReInfarction'] == 1 and x['LengthDapt_months'] < x['Months_reAMI']:
        x['ReInfarctionAfterDapt'] = 1
    return x['ReInfarctionAfterDapt']
```

In [ ]:

```
%%time
df['BleedingDapt'] = 0
df['BleedinfAfterDapt'] = 0
df['ReInfarctionDapt'] = 0
df['ReInfarctionAfterDapt'] = 0

# We run the entire functions in our database
df['BleedingDapt'] = df.apply(sangradoDapt, axis = 1)
df['BleedinfAfterDapt'] = df.apply(sangradoAfterDapt, axis = 1)
df['ReInfarctionDapt']= df.apply(reamiDapt, axis = 1)
df['ReInfarctionAfterDapt']= df.apply(reamiAfterDapt, axis = 1)
```

---

5. Create new variables that classified infarction/bleeding at 6 months, 12 months and 24 months.

In [ ]:

```
# WHILE THE MEDICATION
# Create a new variable, to study if he had a heart attack or bleeding before the age of 6 months.
df['reAMIless6Dapt'] = 0
df.loc[(df['Months_reAMI'] <= 6) & (df['ReInfarctionDapt']==1), 'reAMIless6Dapt'] = 1

df['bleedingless6Dapt'] = 0
df.loc[(df['Months_Bleeding'] <= 6) & (df['BleedingDapt']==1), 'bleedingless6Dapt'] = 1

# Create a new variable, to study if he had a heart attack or bleeding before the age of 12 months
df['reAMIless12Dapt'] = 0
df.loc[(df['Months_reAMI'] <= 12) & (df['ReInfarctionDapt']==1), 'reAMIless12Dapt'] = 1

df['bleedingless12Dapt'] = 0
df.loc[(df['Months_Bleeding'] <= 12) & (df['BleedingDapt']==1), 'bleedingless12Dapt'] = 1

# Create a new variable, to study if he had a heart attack or bleeding before the age of 24 months
df['reAMIless24Dapt'] = 0
df.loc[(df['Months_reAMI'] <= 24) & (df['ReInfarctionDapt']==1), 'reAMIless24Dapt'] = 1

df['bleedingless24Dapt'] = 0
df.loc[(df['Months_Bleeding'] <= 24) & (df['BleedingDapt']==1), 'bleedingless24Dapt'] = 1
```

Since the study will be performed during medication, it is not necessary to run the following block of commands

In [ ]:

```
# # Creamos una nueva variable, para estududiar si tuvo infarto o sangrado antes de los 6 meses.
# df['reAMIless6'] = 0
# df.loc[(df['Months_reAMI'] <= 6) & (df['ReInfarction']==1), 'reAMIless6'] = 1
```

```
# df['bleedingless6'] = 0
# df.loc[(df['Months_Bleeding'] <= 6) & (df['Bleeding']==1), 'bleedingless6'] = 1

# # Creamos una nueva variable, para estududiar si tuvo infarto o sangrado antes de los 12 meses.
# df['reAMIless12'] = 0
# df.loc[(df['Months_reAMI'] <= 12) & (df['ReInfarction']==1), 'reAMIless12'] = 1

# df['bleedingless12'] = 0
# df.loc[(df['Months_Bleeding'] <= 12) & (df['Bleeding']==1), 'bleedingless12'] = 1

# # Creamos una nueva variable, para estududiar si tuvo infarto o sangrado antes de los 24 meses.
# df['reAMIless24'] = 0
# df.loc[(df['Months_reAMI'] <= 24) & (df['ReInfarction']==1), 'reAMIless24'] = 1

# df['bleedingless24'] = 0
# df.loc[(df['Months_Bleeding'] <= 24) & (df['Bleeding']==1), 'bleedingless24'] = 1
```

Export Preprocess Database

In [ ]:

```
# df.to_excel('../../data/PACS_DAPT_preprocess.xlsx')
df.to_csv('../../data/PACS_DAPT_preprocess.csv')
```

2_Model

In [ ]:

```
# Import librarys

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Algorthms / graphics
from sklearn.metrics import roc_auc_score, confusion_matrix, accuracy_score
from sklearn import preprocessing
from sklearn.model_selection import cross_val_score
# from sklearn.linear_model import LogisticRegression
# from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from yellowbrick.classifier import ClassificationReport, DiscriminationThreshold, ROCAUC, Confusi
onMatrix, PrecisionRecallCurve

%pylab inline
```

In [ ]:

```
# Import data

df = pd.read_csv('../../data/PACS_DAPT_preprocess.csv')
df = df.drop('Unnamed: 0', axis=1)
```

In [ ]:

```
# Variables predicting our Dataset, those variables that don't have empty values (that's why BMI is mis
sing)
var_pred = ['Age', 'Sex', 'Diabetes', 'Hypertension', 'Dyslipemia', 'PeripheralArteryDisease',
        'PriorMyocardialInfarction', 'PriorPCI', 'PriorCABG', 'PriorStroke', 'Cancer', 'Prior_Bleeding',
        'STEMI', 'UnstableAngina', 'Killip2orMore', 'LVEFless40', 'LVEFless50', 'LVEFafter50', 'Hem
oglobin_Admission',
        'Creatinine_Admission', 'Femoral_Access', 'Multivessel', 'DrugElutingStent', 'NoStent',
        'Revascularization_Complete', 'ReAMI_inhospital', 'Bleeding_inhospital',
        'Clopidogrel', 'Ticagrelor', 'Prasugrel', 'OralAnticoagulation', 'Betablockers', 'ACEI_ARB', 'Stati
n']
```

Hemorragia¶

In [ ]:

```
X_train = df.loc[(df.Registro == 'BLEEMACS')|(df.Registro == 'RENAMI'), var_pred]
X_test = df.loc[(df.Registro == 'Vigo_Arritxaca'), var_pred]

y_train = df.loc[(df.Registro == 'BLEEMACS')|(df.Registro == 'RENAMI'), 'bleedingless12Dapt']
y_test = df.loc[(df.Registro == 'Vigo_Arritxaca'), 'bleedingless12Dapt']
```

Random Forest¶

In [ ]:

```
# cross-validation

clf = RandomForestClassifier(max_depth= 2, max_features= "auto",random_state=1234)
scores = cross_val_score(clf, X_train, y_train, cv=10, scoring='roc_auc')
print(scores ,'\n\n')
print(scores.mean())
```

In [ ]:

```
clf = clf.fit(X_train, y_train)
y_prob = clf.predict_proba(X_test)
print(roc_auc_score(y_test, y_prob[:,1]))
```

In [ ]:

```
clf.feature_importances_
```

In [ ]:

```
visualizer = ROCAUC(clf, classes=["No bleeding", "bleeding"])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.poof();
```

In [ ]:

```
# visualizer = PrecisionRecallCurve(clf, classes=['no bleeding', 'bleeding'])

# visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
# visualizer.score(X_test, y_test)        # Evaluate the model on the test data
# visualizer.show()
```

In [ ]:

```
visualizer = ConfusionMatrix(clf, classes=['no bleeding', 'bleeding'])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.show()
```

In [ ]:

```
visualizer = ClassificationReport(clf, classes=['no bleeding', 'bleeding'])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.show()
```

Reinfarto¶

In [ ]:

```
X_train = df.loc[(df.Registro == 'BLEEMACS')|(df.Registro == 'RENAMI'), var_pred]
X_test = df.loc[(df.Registro == 'Vigo_Arritxaca'), var_pred]
```

```
y_train = df.loc[(df.Registro == 'BLEEMACS')|(df.Registro == 'RENAMI'), 'reAMIless12Dapt']
y_test = df.loc[(df.Registro == 'Vigo_Arritxaca'), 'reAMIless12Dapt']
```

Random Forest¶

In [ ]:

```
# cross-validation

clf = RandomForestClassifier(max_depth= 2, max_features= 1,random_state=1234)
scores = cross_val_score(clf, X_train, y_train, cv=10, scoring='roc_auc')
print(scores ,'\n\n')
print(scores.mean())
```

In [ ]:

```
clf = clf.fit(X_train, y_train)
y_prob = clf.predict_proba(X_test)
print(roc_auc_score(y_test, y_prob[:,1]))
```

In [ ]:

```
clf.feature_importances_
```

In [ ]:

```
visualizer = ROCAUC(clf, classes=["No reAMI", "reAMI"])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.poof();
```

In [ ]:

```
# visualizer = PrecisionRecallCurve(clf, classes=['no reAMI', 'reAMI'])

# visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
# visualizer.score(X_test, y_test)        # Evaluate the model on the test data
# visualizer.show()
```

In [ ]:

```
visualizer = ConfusionMatrix(clf, classes=['no reAMI', 'reAMI'])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.show()
```

In [ ]:

```
visualizer = ClassificationReport(clf, classes=['no reAMI', 'reAMI'])

visualizer.fit(X_train, y_train)        # Fit the training data to the visualizer
visualizer.score(X_test, y_test)        # Evaluate the model on the test data
visualizer.show()
```