Universitat Autònoma de Barcelona

Informe de practica de Aprendizaje computacional:

Clasificación

Luis Fernando Paz Galeano	1567369
Miguel Del Arco Márquez	1566698

Índice

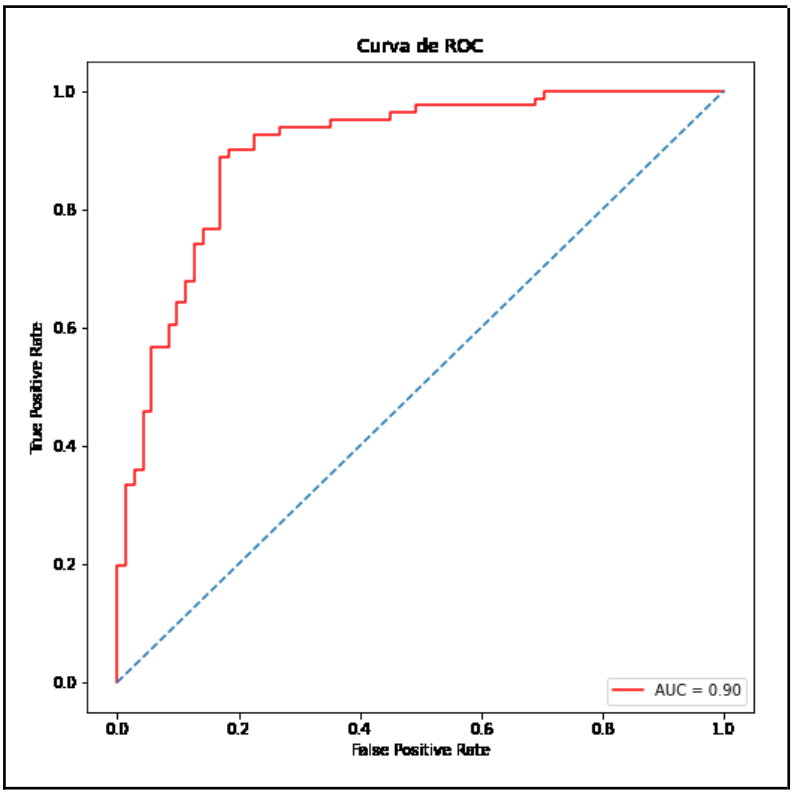
Apartado B comparación de modelos

A continuación se procede a mostrar los resultados de los diferentes análisis realizados, para ello hemos seleccionado tres diferentes modelos, regresión logística, máquinas de vectores de soporte, y finalmente Random forest.

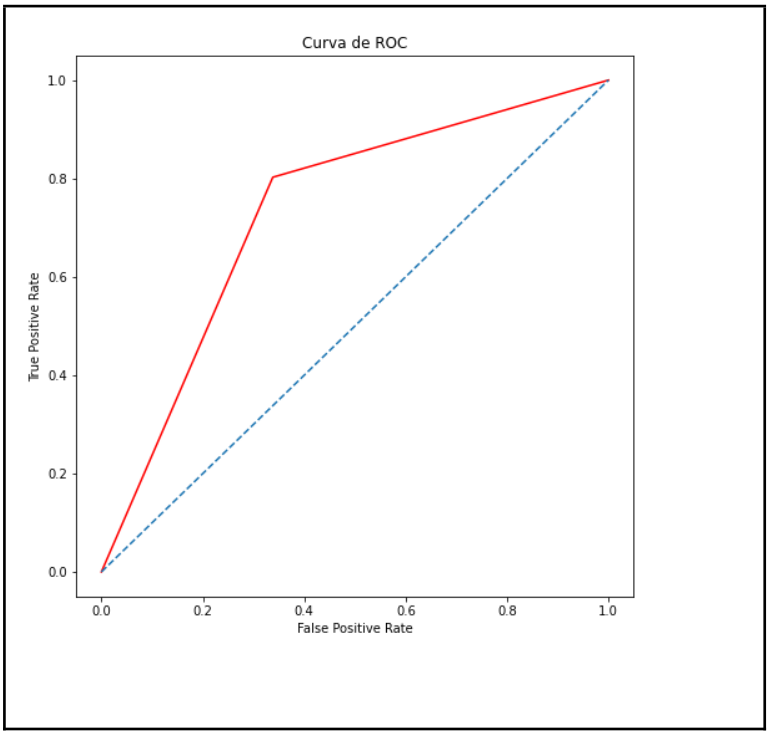
Para la comparación de las diferentes curvas de ROC, hemos seleccionado los parámetros necesarios para poder hacer una comparativa equitativa entre los diferentes modelos es decir hemos partido de la base que hemos seleccionado aquellos parámetros que dan lugar a que se produzca la mayor precisión.

Mencionar también que durante la realización del apartado A hemos ido testeando también para cada uno de los modelos con diferentes valores en los parámetros, posteriormente proceder a explicar cuáles fueron los resultados obtenidos.

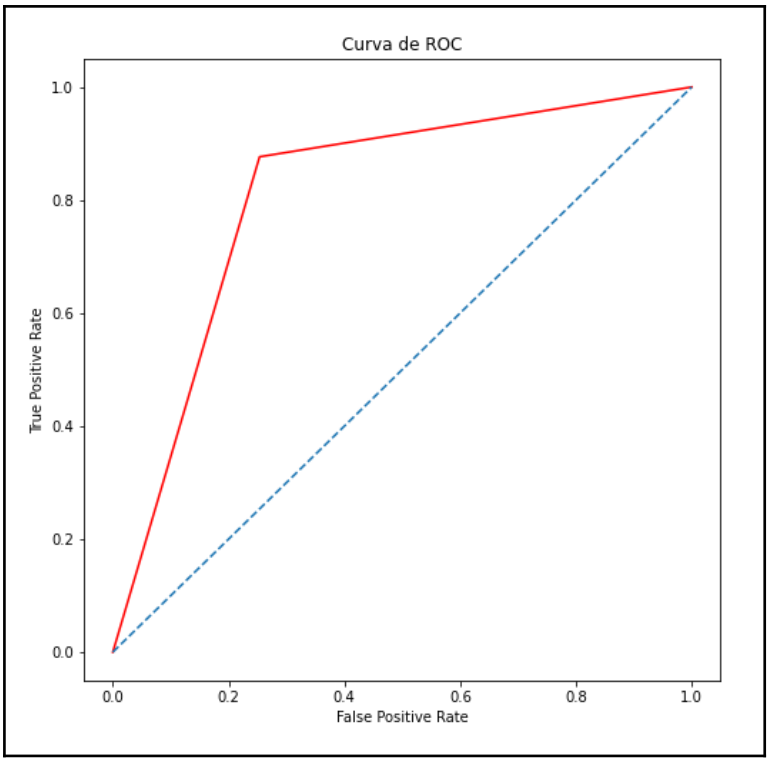
Modelo de Regresión logística curva de ROC



Modelo de máquinas de vectores de soporte curva de ROC



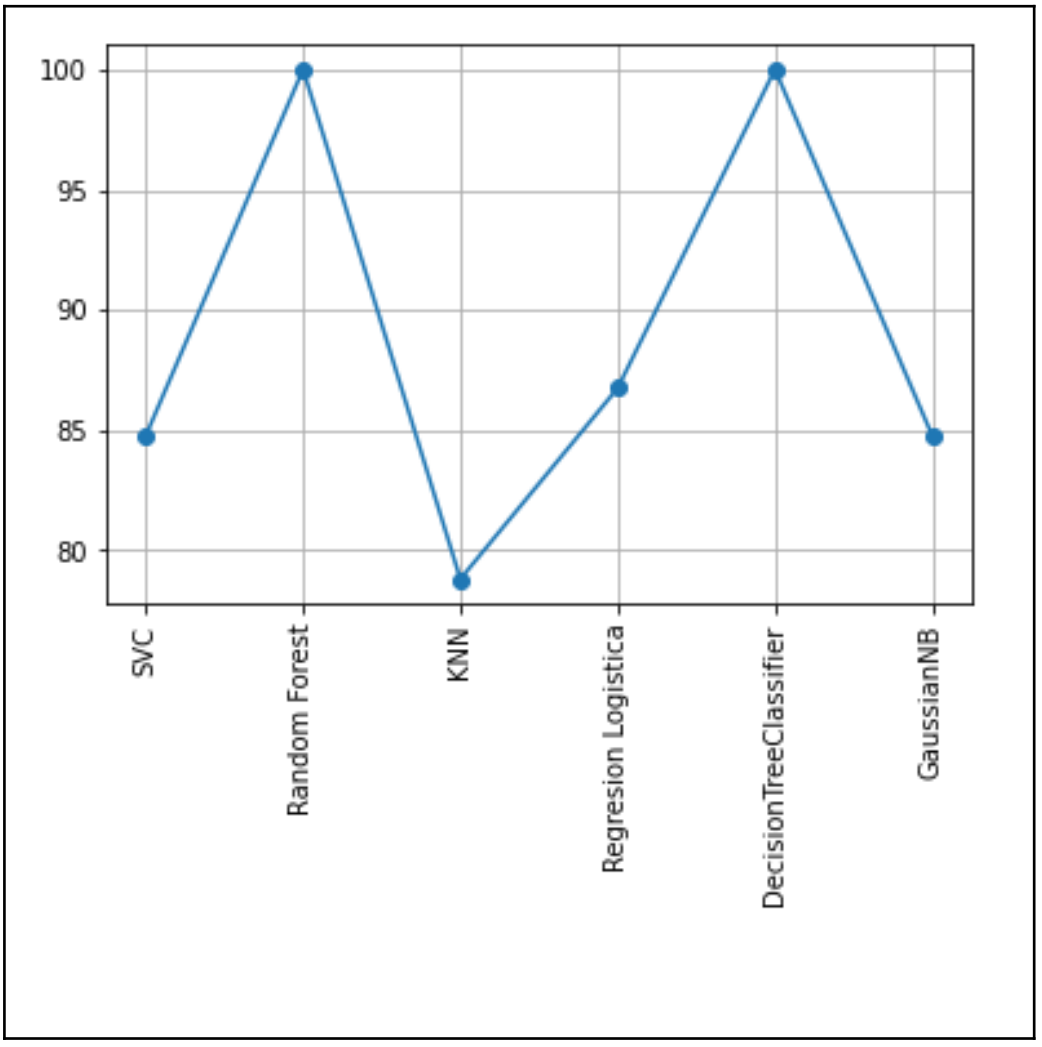
Modelo de Random Forrest curva de ROC



Resultados obtenidos en base a la precisión del modelo

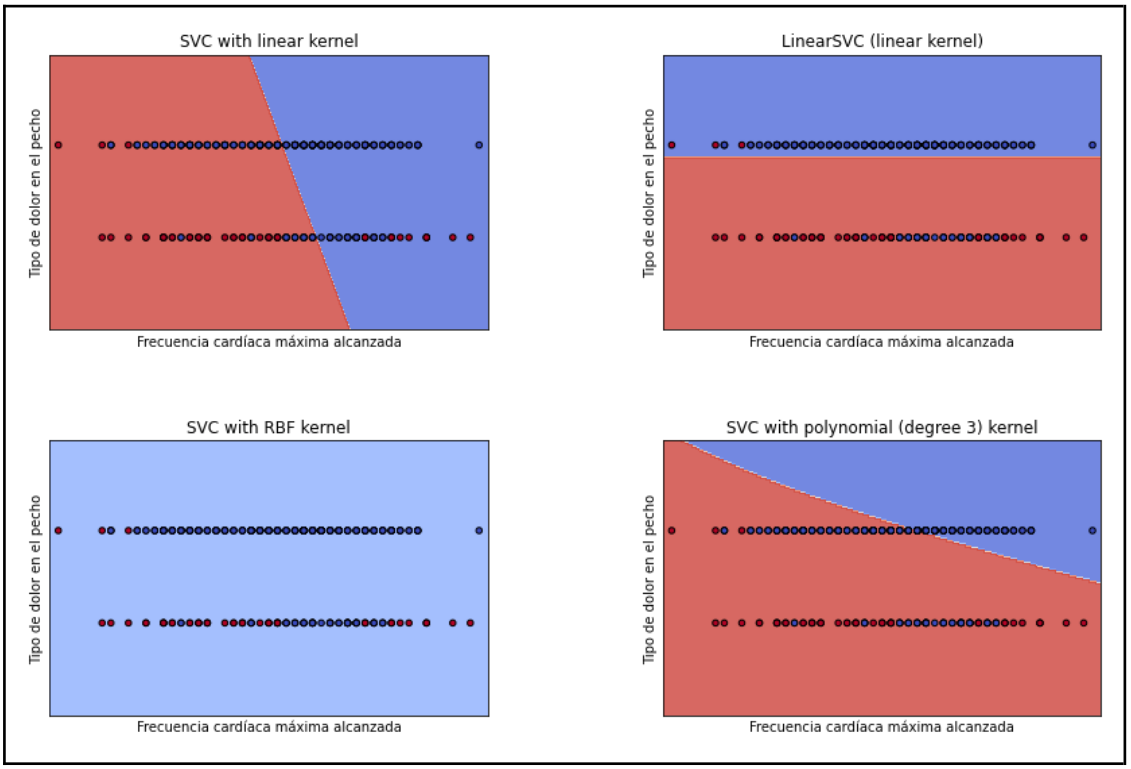
Precision = SVC : 84.77 %
Precision = Random Forest : 100.0 %
Precision = KNN : 78.81 %
Precisión = Regresión Logística : 86.75 %
Precision = DecisionTreeClassifier : 100.0 %
Precision = GaussianNB : 84.77 %

Resultados obtenidos en base a la precisión del modelo gráfico.

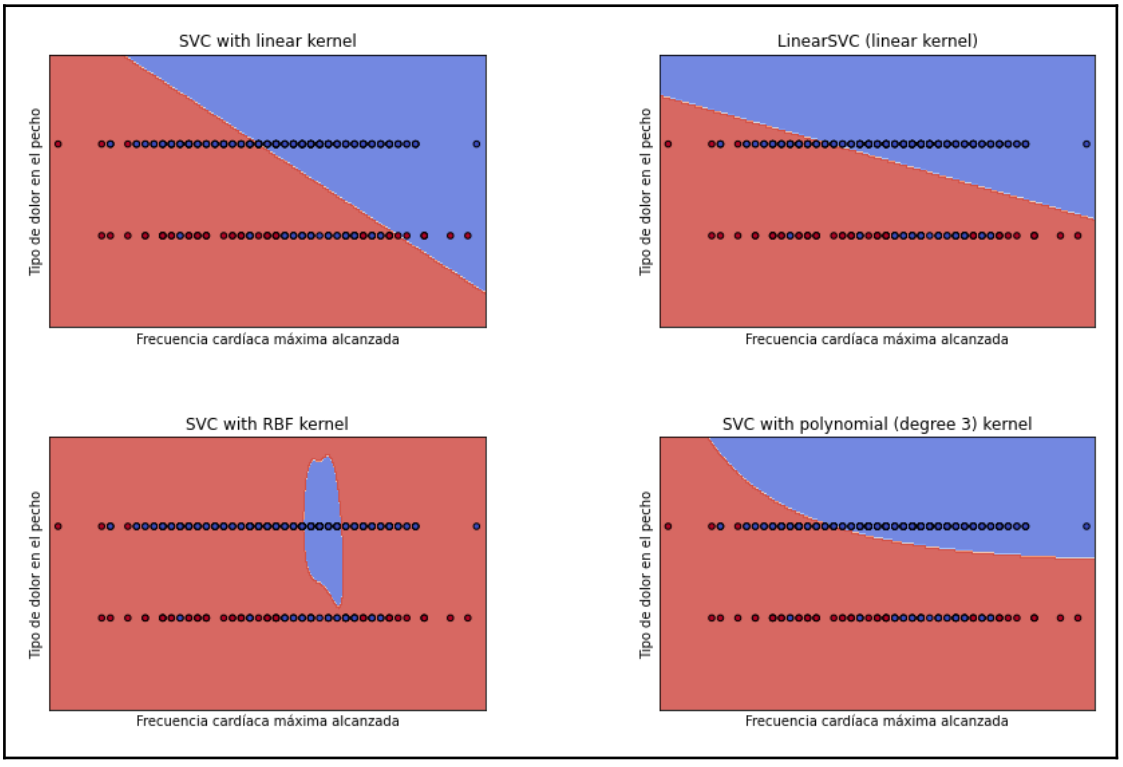


Calificaciones de los modelos en base al valor de la ‘C’ utilizando máquinas de vectores de soporte

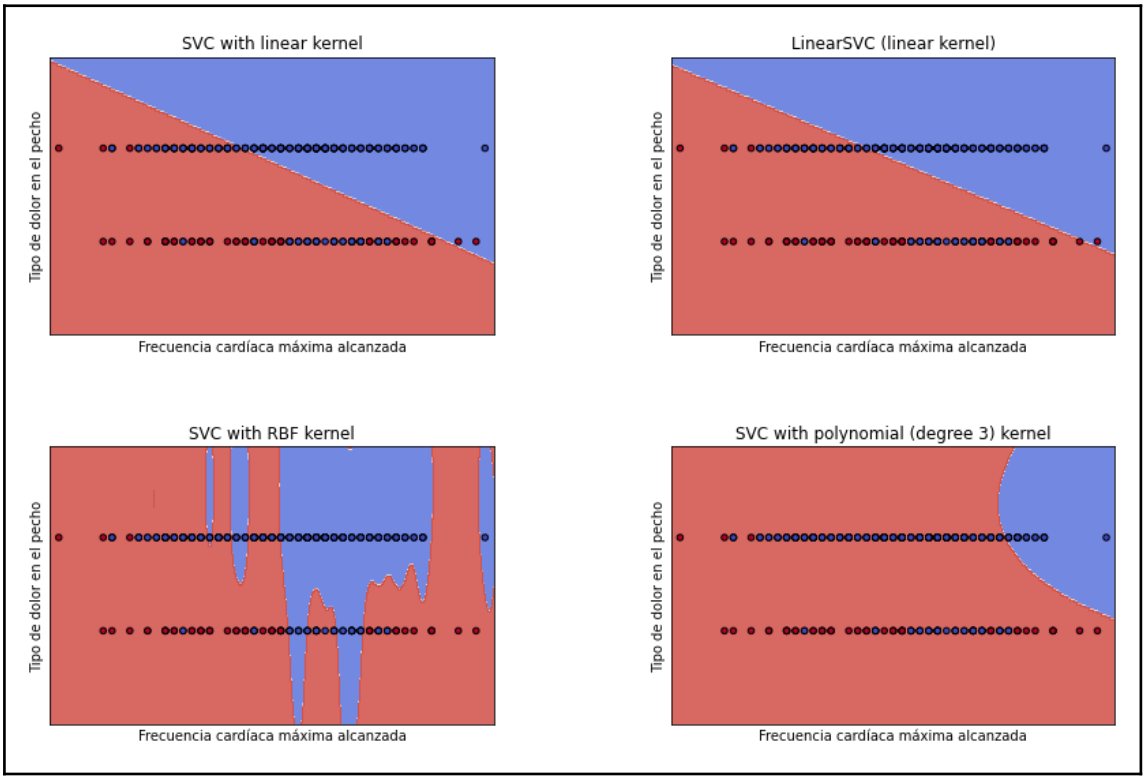
C=0.01



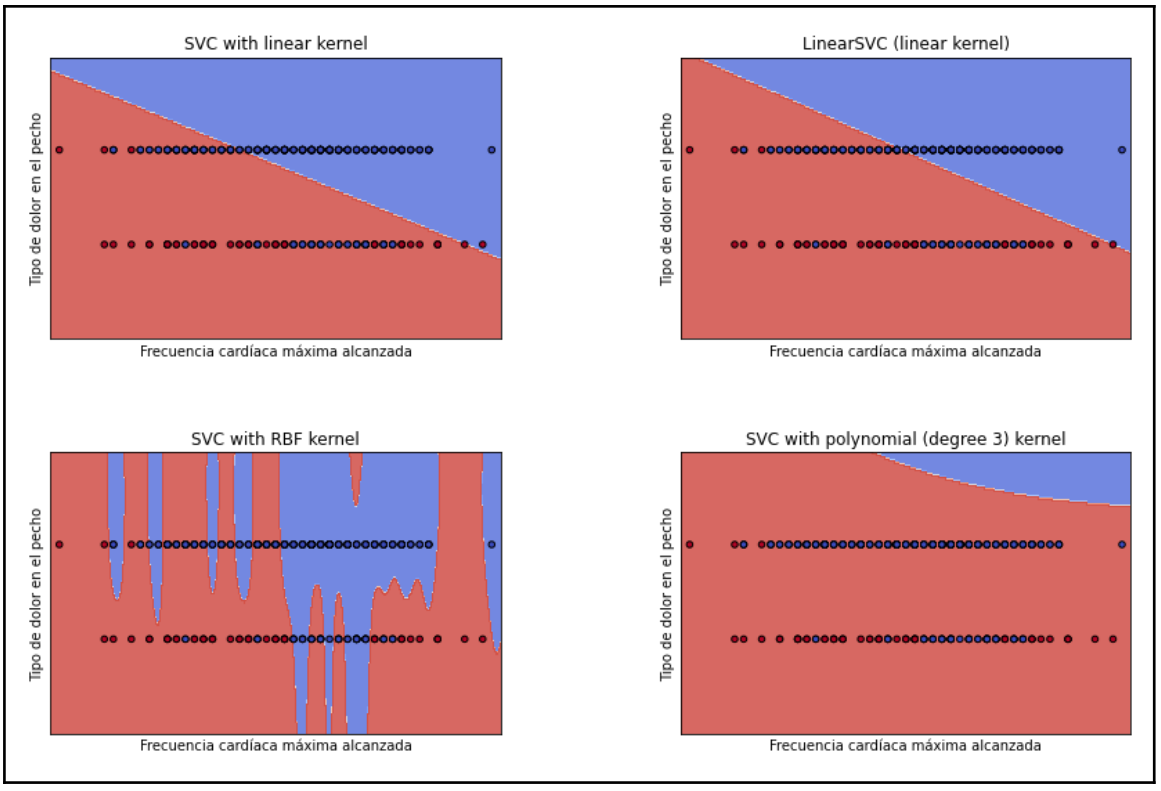
C=0.1



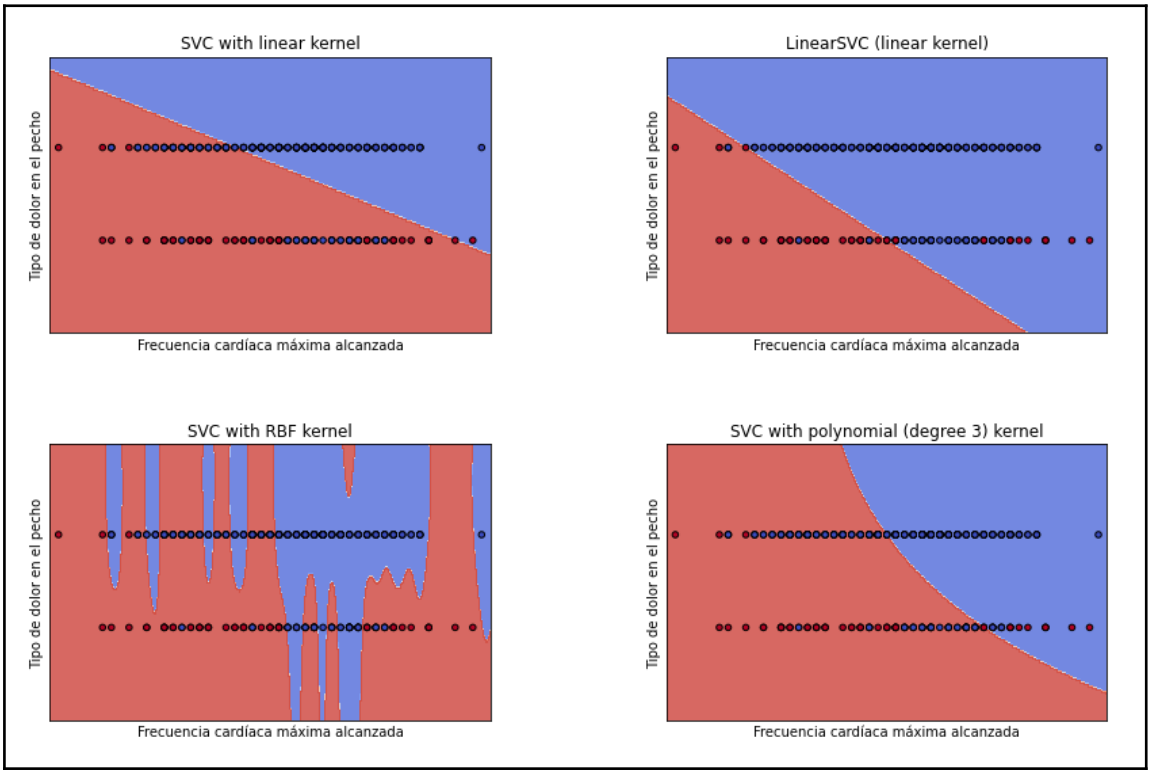
C=1.0


SVC with linear kernel
LinearSVC (linear kernel)
SVC with RBF kernel
SVC with polynomial (degree 3) kernel
Tipo de dolor en el pecho
Frecuencia cardíaca máxima alcanzada

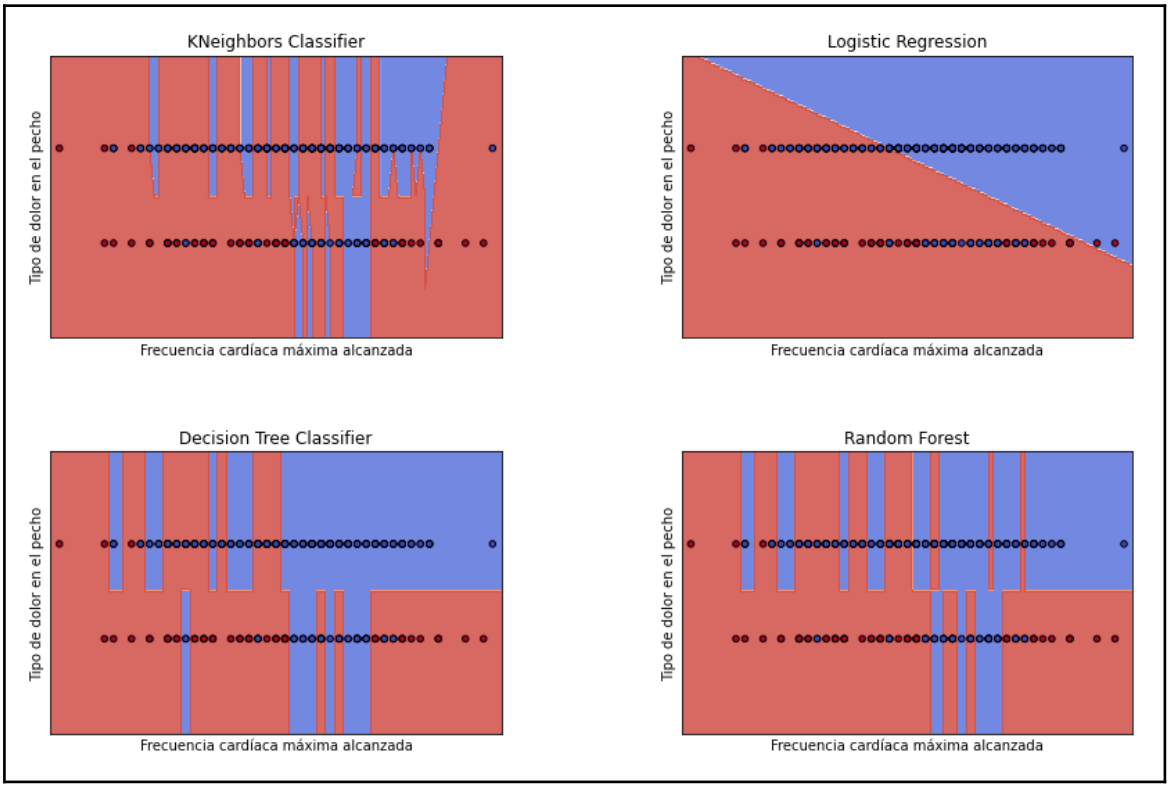
C=100


SVC with linear kernel
LinearSVC (linear kernel)
SVC with RBF kernel
SVC with polynomial (degree 3) kernel
Tipo de dolor en el pecho
Frecuencia cardíaca máxima alcanzada

C=1000



Comparación de varios modelos



Apartado A clasificación numérica

1. EDA (exploratory data analysis)

Respuesta a preguntas planteadas.

¿Cuántos atributos tiene su base de datos?

La base de datos cuenta con un total de 13 atributos.

¿Qué tipo de atributos tienes?

Todos los datos que se encuentran en la base de datos son de carácter categórico.

Como es el target, ¿cuántas categorías diferentes existen?

El target o output solo tiene una categoría, es decir en este caso predecir si un paciente padece de una enfermedad del corazón o no.

¿Puede ver alguna correlación entre X y y?

Sí que existe una correlación ya que los datos son muy determinantes para obtener una calificación final.

¿Están balanceadas las etiquetas (distribución similar entre categorías)? ¿Crees que puede afectar a la clasificación su distribución?

En este caso los diferentes atributos que presenta la base de datos tienen una correlación relativamente baja entre sí, por lo cual son ideales para poder hacer una clasificación.

Información sobre el DataSet

Puede consultar la información referente al DataSet Heart Attack Analysis and Prediction Dataset.
El dataset recoge información referente a un sin número de pacientes que han sido evaluados y en cada uno de los atributos de la base de datos ha sido registrado los diferentes valores cómo puede ser edad, sexo, presión cardíaca, nivel de colesterol, nivel de azúcar en sangre etc.
la variable objetivo que en este caso es de la Output, nos indica si dicho paciente ha sufrido algún ataque al corazón, por lo cual el objetivo de nuestra clasificación será precisamente pero decir si en base a un conjunto de entradas es decir variables independientes, obtener aquella variable dependiente que nos indicará o clasificará en base a ese conjunto de entradas, si la persona tiene más posibilidades de ataque cardíaco o no.

Información de los atributos

- *Age : Age of the patient*
- *Sex : Sex of the patient*
- *exang: exercise induced angina (1 = yes; 0 = no)*
- *ca: number of major vessels (0-3)*
- *cp : Chest Pain type chest pain type*
  - *Value 1: typical angina*
  - *Value 2: atypical angina*
  - *Value 3: non-anginal pain*
  - *Value 4: asymptomatic*
- *trtbps : resting blood pressure (in mm Hg)*
- *chol : cholestoral in mg/dl fetched via BMI sensor*
- *fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)*
- *rest_ecg : resting electrocardiographic results*
  - *Value 0: normal*
  - *Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)*
  - *Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria*
- *thalach : maximum heart rate achieved*
- *target : 0= **less chance of heart attack** 1= **more chance of heart attack***

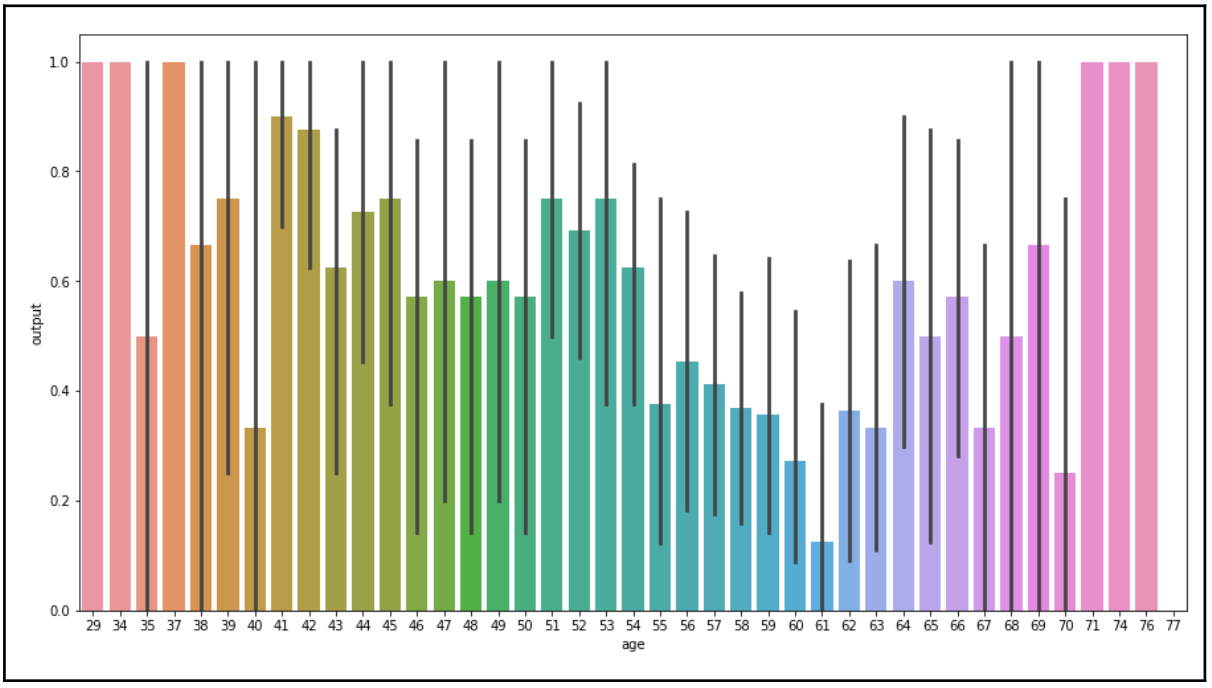
Estadística de los datos

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
count	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000	303.000
mean	54.366	0.683	0.967	131.624	246.264	0.149	0.528	149.647	0.327	1.040	1.399	0.729	2.314	0.545
std	9.082	0.466	1.032	17.538	51.831	0.356	0.526	22.905	0.470	1.161	0.616	1.023	0.612	0.499
min	29.000	0.000	0.000	94.000	126.000	0.000	0.000	71.000	0.000	0.000	0.000	0.000	0.000	0.000
25%	47.500	0.000	0.000	120.000	211.000	0.000	0.000	133.500	0.000	0.000	1.000	0.000	2.000	0.000
50%	55.000	1.000	1.000	130.000	240.000	0.000	1.000	153.000	0.000	0.800	1.000	0.000	2.000	1.000
75%	61.000	1.000	2.000	140.000	274.500	0.000	1.000	166.000	1.000	1.600	2.000	1.000	3.000	1.000
max	77.000	1.000	3.000	200.000	564.000	1.000	2.000	202.000	1.000	6.200	2.000	4.000	3.000	1.000

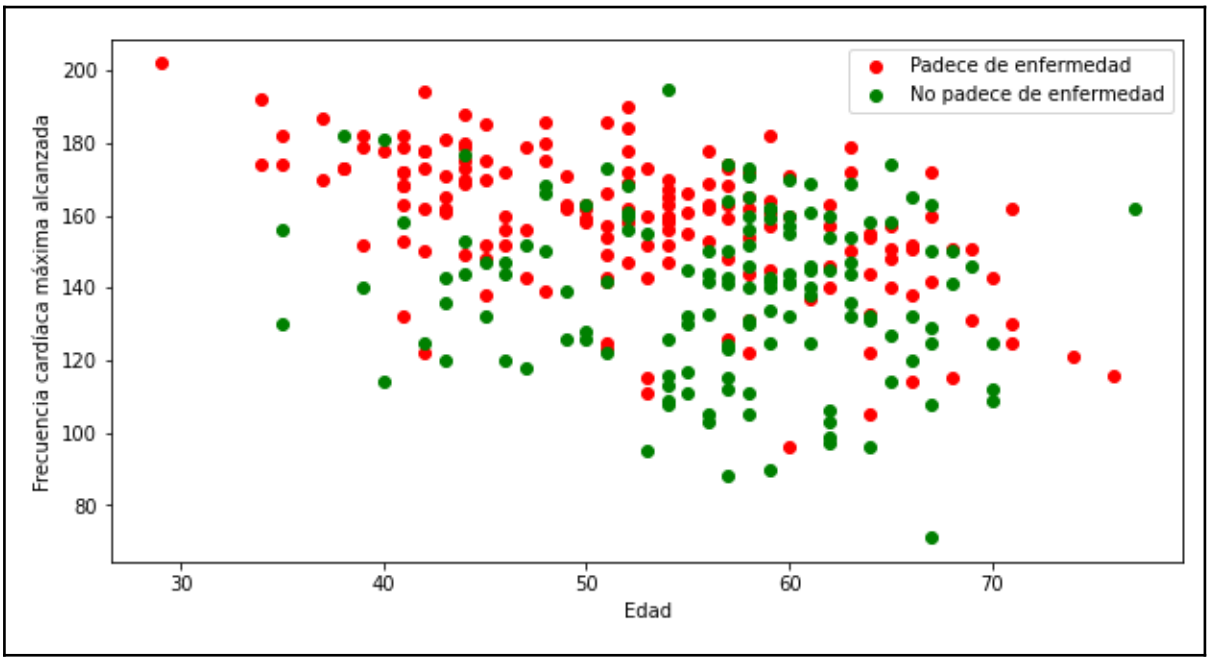
Exploración de los datos

Con el objetivo de poder comprender la distribución y relación existente entre los datos de nuestra base de datos hemos procedido a hacer un análisis y estudio de cada uno de los atributos, a continuación procederemos a mostrar los resultados obtenidos.

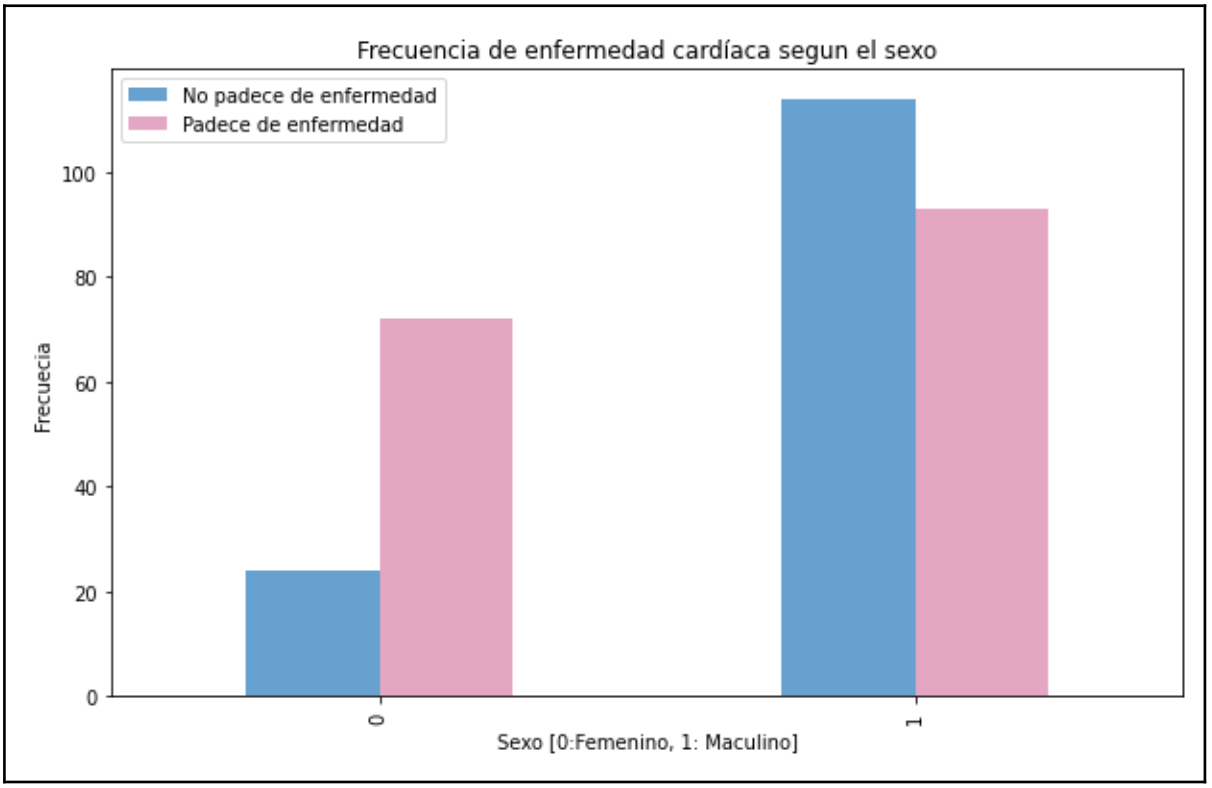
Posibilidad de padecer un ataque cardíaco en base a la edad.



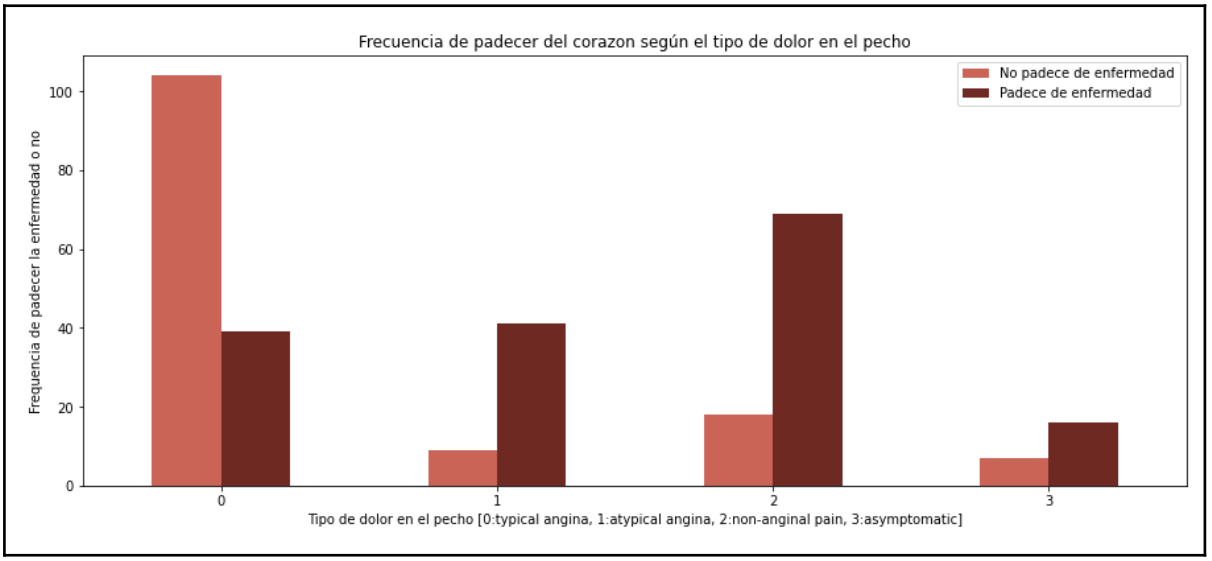
Frecuencia cardíaca máxima alcanzada en base a la probabilidad de sufrir un ataque cardíaco.



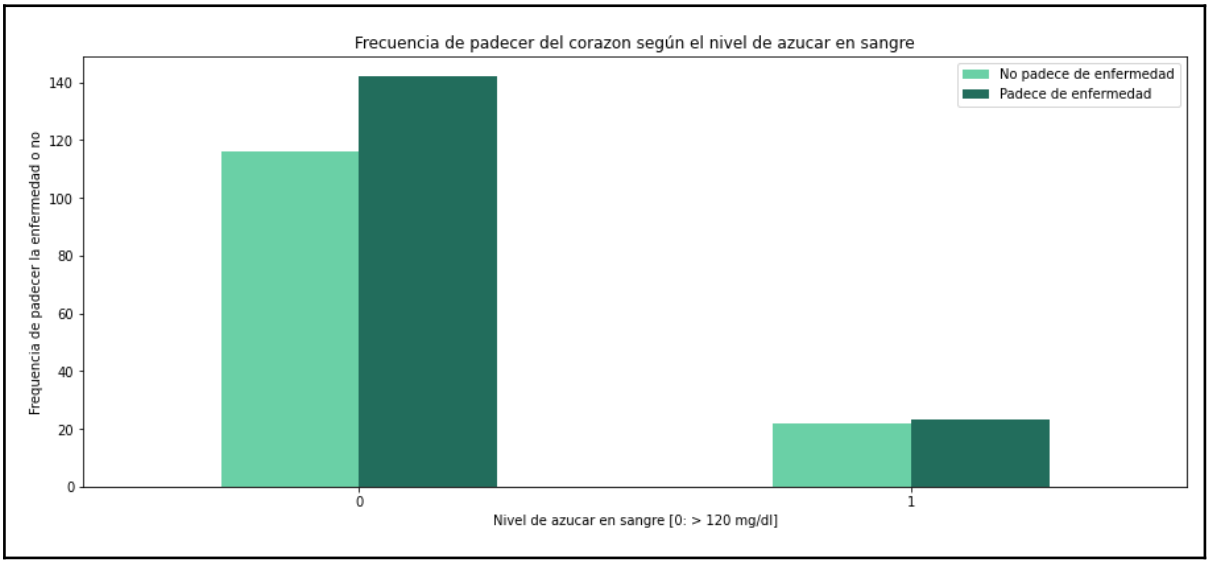
Frecuencia de padecimiento de ataque cardíaco según el sexo



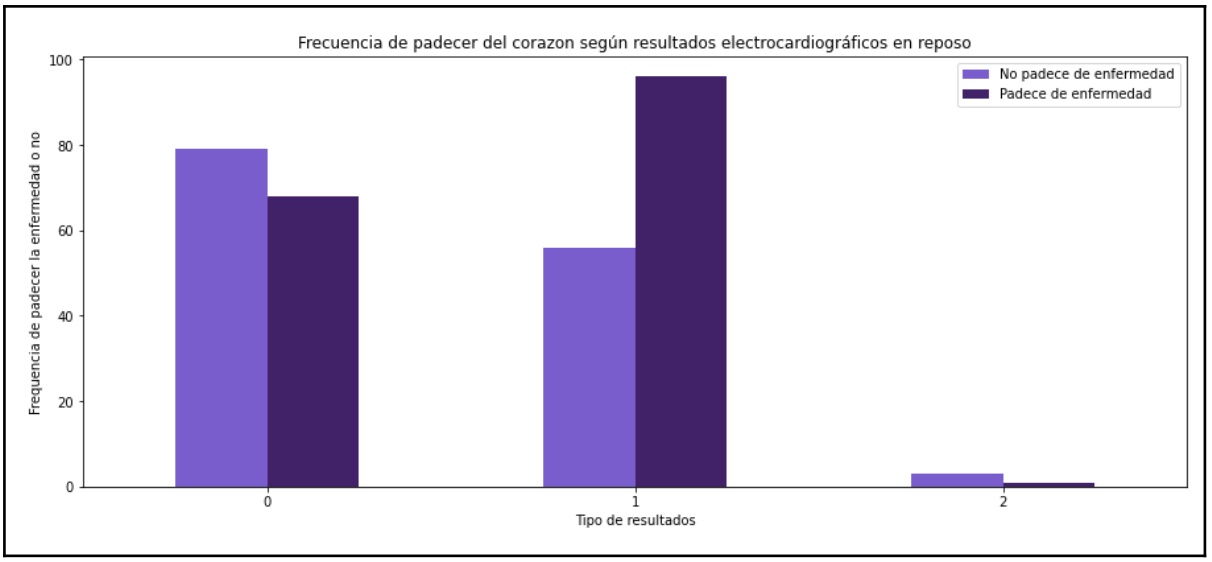
Frecuencia de parecer un ataque cardíaco según el tipo de dolor de pecho



Frecuencia de padecer un ataque cardíaco según el nivel de azúcar en sangre

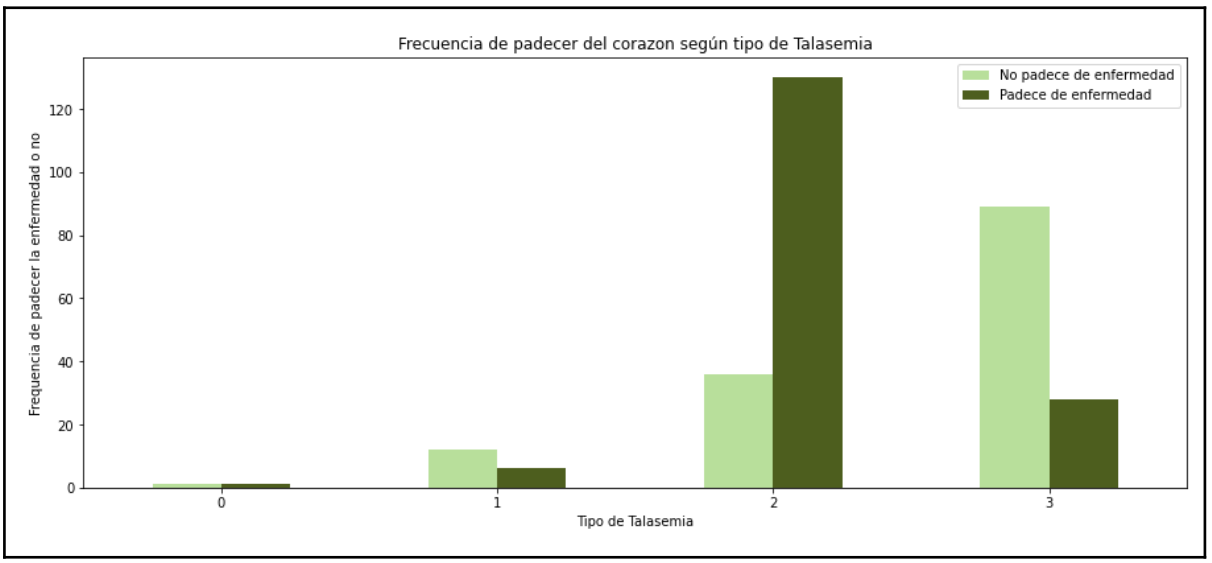


Frecuencia de padecer un ataque cardíaco según resultados electrocardiográficos en reposo.



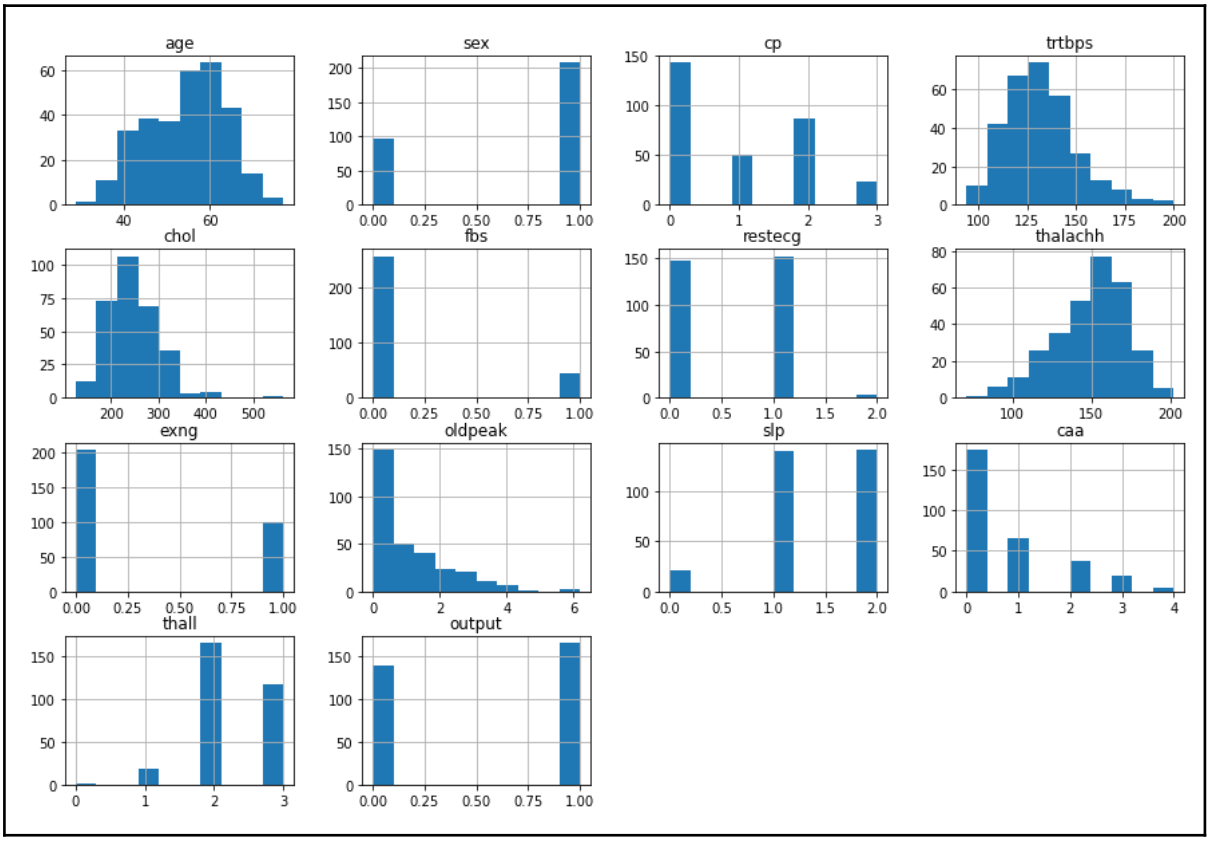
Frecuencia de padecer un ataque cardíaco según el tipo de talasemia

Talasemia: La talasemia es un trastorno sanguíneo hereditario que hace que tu cuerpo tenga menos hemoglobina de lo normal.



Análisis de multivariables

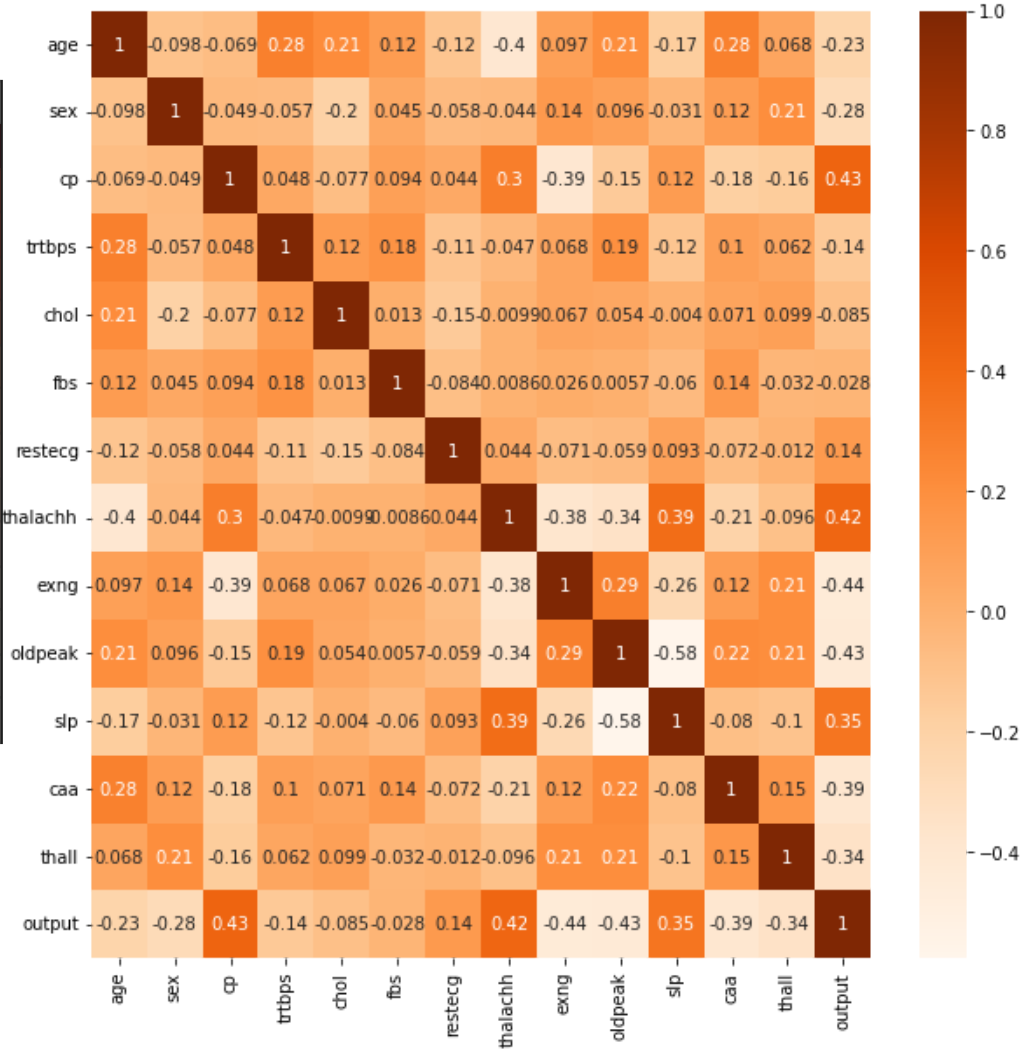
Distribución de los datos.



Correlación de los atributos

Cómo podemos observar existe una alta correlación al menos en tres de los atributos en referencia a la variable que se precisa de terminar estos atributos son, ‘Tipo de dolor de pecho’,’Frecuencia máxima alcanzada ‘ y ’La pendiente del pico ejercicio segmento ST’. Así también podemos observar una correlación entre los diferentes atributos, y también podemos predecir que dichos atributos serán significativos al momento de realizar la clasificación.

	output
output	**1.000000**
cp	**0.433798**
thalachh	**0.421741**
slp	**0.345877**
restecg	**0.137230**
fbs	**-0.028046**
chol	**-0.085239**
trtbps	**-0.144931**
age	**-0.225439**
sex	**-0.280937**
thall	**-0.344029**
caa	**-0.391724**
oldpeak	**-0.430696**
exng	**-0.436757**



2. Preprocessing

¿Están los datos normalizados? ¿Habría que hacerlo?

En nuestro caso los datos no estaban normalizados pero sí que mantenían una similitud respecto a los rangos.

En caso de que las normalice, ¿qué tipo de normalización será más adecuada para sus datos?

Hemos normalizado aquellos datos que por su tipo, ya sea bien porque contenían valores muy altos, o bien porque no tenían similitud respecto a rangos con los demás atributos, y la normalización que hemos aplicado ha sido la minimax

¿Tiene muchos datos sin información? ¿Los NaNs en pandas?

En nuestro caso hemos revisado y no teníamos datos nulos y tampoco teníamos datos información.

¿Cómo afecta a la clasificación si las filtramos? ¿Y si las rellena? ¿Cómo lo haría?.

Como hemos mencionado anteriormente no teníamos datos nulos y tampoco hemos hecho ningún experimento añadiendo ningún tipo de valor anormal.

¿Tiene datos categóricos? ¿Cuál sería la codificación con mayor sentido?

Todos los atributos de nuestro DataSet eran datos de tipo categórico.

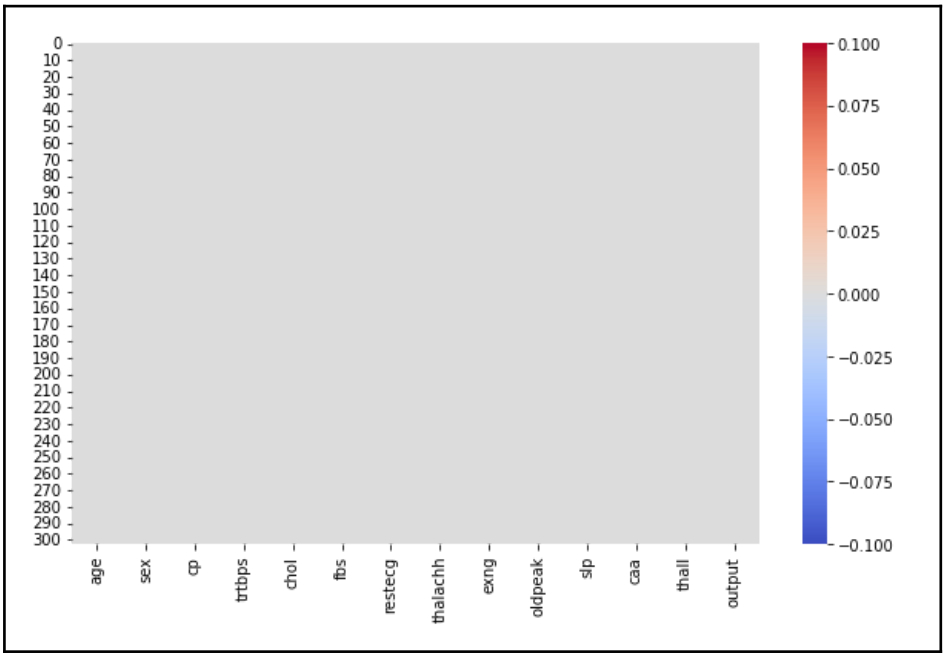
¿Se pueden aplicar `Polynomial Features` para mejorar la clasificación?

La regresión lineal es lineal en los parámetros del modelo y agregar términos polinomiales al modelo puede ser una forma eficaz de permitir que el modelo identifique patrones no lineales.

En nuestro caso no utilizamos dichas herramientas ya que pudimos determinar Existe una clara relación lineal entre las variables independientes y dependiente.

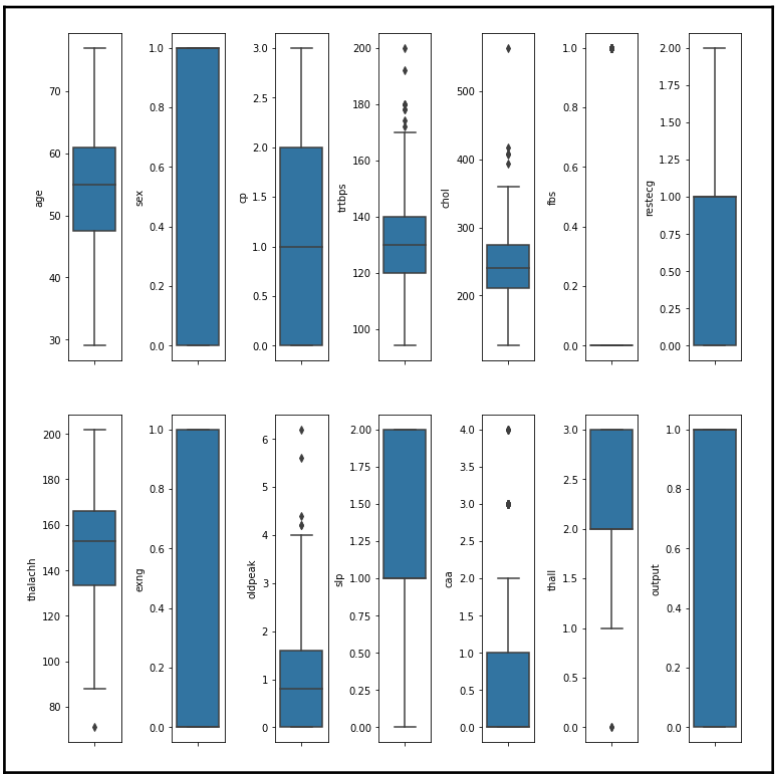
Análisis de elementos nulos.

Hemos procedido a analizar y disponíamos de elementos nulos en nuestro DataSet, en este caso no disponíamos de elementos Nulos pero para realizar una buena clasificación sería necesario hacer un buen procesamiento de los datos.



Así también hemos analizado si teníamos elementos repetidos, en este caso sí que contábamos con elementos repetidos y hemos procedido a eliminarlos.

Análisis de los elementos con valores atípicos.



Normalización de los datos

Hemos normalizado aquellos atributos que consideramos que por sus dimensiones eran necesarios para aplicar dicha normalización, cabe mencionar que todos los atributos de nuestra base de datos son de carácter categórico, hemos normalizado los valores correspondientes a la edad,Frecuencia cardíaca,Nivel de colesterol etc

```
['trtbps', 'chol', 'thalachh', 'oldpeak', 'age']
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	0.708	1	3	0.481	0.244	1	0	0.603	0	0.371	0	0	1	1
1	0.167	1	2	0.340	0.283	0	1	0.885	0	0.565	0	0	2	1
2	0.250	0	1	0.340	0.178	0	0	0.771	0	0.226	2	0	2	1
3	0.562	1	1	0.245	0.251	0	1	0.817	0	0.129	2	0	2	1
4	0.583	0	0	0.245	0.521	0	1	0.702	1	0.097	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	0.583	0	0	0.434	0.263	0	1	0.397	1	0.032	1	0	3	0
299	0.333	1	3	0.151	0.315	0	1	0.466	0	0.194	1	0	3	0
300	0.812	1	0	0.472	0.153	1	1	0.534	0	0.548	1	2	3	0
301	0.583	1	0	0.340	0.011	0	1	0.336	1	0.194	1	1	3	0
302	0.583	0	1	0.340	0.251	0	0	0.786	0	0.000	1	1	2	0

Codificación con One Hot Encoding (Revisarlo en el código)

Aún no me queda claro cómo funciona.

3. Model Selection

¿Qué modelos ha considerado?
Hemos considerado los siguientes modelos, regresión logística, máquina de vectores de soporte con los diferentes tipos de kernel, random forest, knn.
¿Cuál cree que será el más preciso?
El kernel de tipo rbf.
¿Cuál será el más rápido?
Asumimos que por su simplicidad el kernel de tipo linear.

Debido a que en nuestro DataSet había una clara diferencia entre la variable dependiente y el conjunto de variables independientes no hemos tenido que modificar mucho los datos, lo único que hemos realizado es una normalización del conjunto de las variables que se ha mencionado anteriormente.
Por lo cual tenemos lo siguiente:

conjunto de variables independientes

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall
0	0.708	1	3	0.481	0.244	1	0	0.603	0	0.371	0	0	1
1	0.167	1	2	0.340	0.283	0	1	0.885	0	0.565	0	0	2
2	0.250	0	1	0.340	0.178	0	0	0.771	0	0.226	2	0	2
3	0.562	1	1	0.245	0.251	0	1	0.817	0	0.129	2	0	2
4	0.583	0	0	0.245	0.521	0	1	0.702	1	0.097	2	0	2

Variable dependiente

output
1
1
1
1
1
...
0
0
0
0
0

Diferentes modelos probados y resultados obtenidos.

Durante las evaluaciones con diferentes modelos correspondientes hemos ido probando diferentes valores para ver los diferentes resultados en base a la variación de los mismos.

Regresión logística

Valores adoptados para el atributo C.

```
c_list=[0.001,0.01,0.1,1.0,2.0]
```

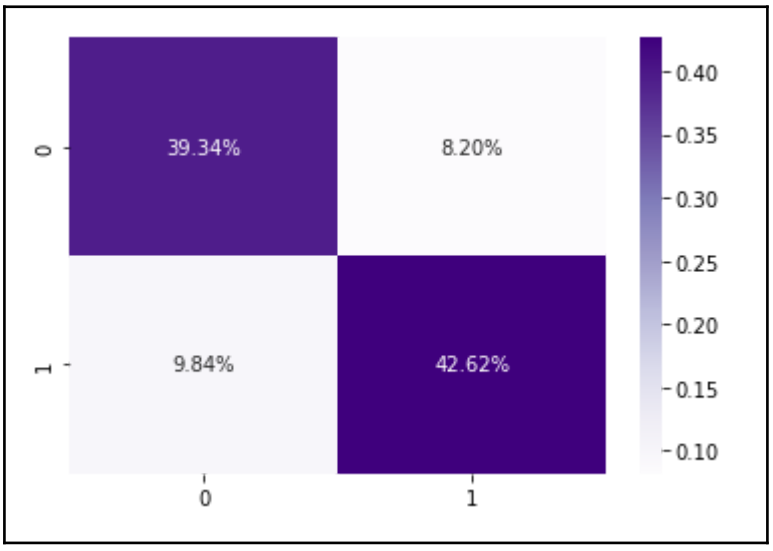
Valores para fit_intercept.

```
f_inter=[False,True]
```

Los resultados obtenidos de la regresión logística fueron los siguientes:

	Modelo	Precision	C	fit_intercept	penalty	tol
2	Regresion Logistica	0.869	0.010	False	l2	0.001
0	Regresion Logistica	0.852	0.001	False	l2	0.001
3	Regresion Logistica	0.852	0.010	True	l2	0.001
6	Regresion Logistica	0.852	1.000	False	l2	0.001
5	Regresion Logistica	0.836	0.100	True	l2	0.001
8	Regresion Logistica	0.836	2.000	False	l2	0.001
4	Regresion Logistica	0.820	0.100	False	l2	0.001
7	Regresion Logistica	0.820	1.000	True	l2	0.001
9	Regresion Logistica	0.820	2.000	True	l2	0.001

Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión.



Máquina de vectores de soporte

Valores adoptados para el atributo C.

```
c_list=[0.001,0.01,0.1,1.0,2.0]
```

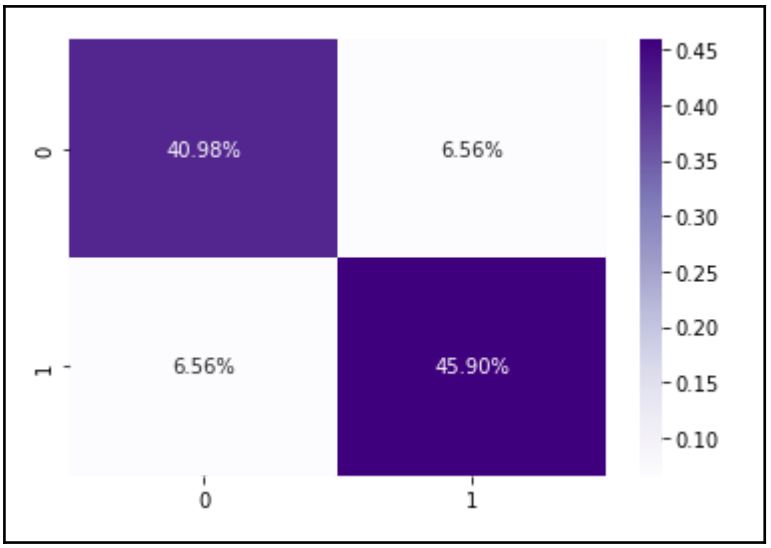
Valores de gamma

```
gammas=['scale','auto']
```

Resultados obtenidos para máquina de vectores de soporte con un kernel de tipo RBF

	Modelo	Precision	C	probability	gamma	tol
4	SVG_rbf	0.869	0.100	True	scale	0.001
5	SVG_rbf	0.869	0.100	True	auto	0.001
6	SVG_rbf	0.869	1.000	True	scale	0.001
7	SVG_rbf	0.869	1.000	True	auto	0.001
8	SVG_rbf	0.869	2.000	True	scale	0.001
9	SVG_rbf	0.869	2.000	True	auto	0.001
0	SVG_rbf	0.525	0.001	True	scale	0.001
1	SVG_rbf	0.525	0.001	True	auto	0.001
2	SVG_rbf	0.525	0.010	True	scale	0.001
3	SVG_rbf	0.525	0.010	True	auto	0.001

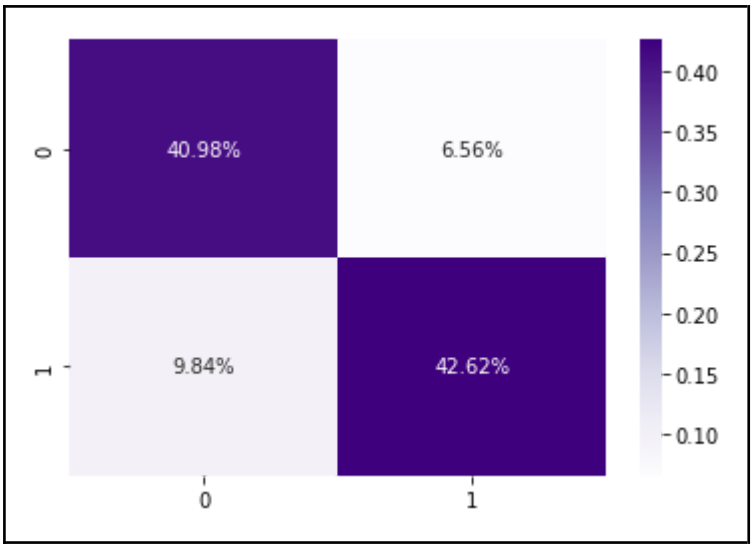
Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión.



Resultados obtenidos para máquina de vectores de soporte con un kernel de tipo linear

	Modelo	Precision	C	probability	gamma	tol
0	SVG_linear	0.836	0.001	True	scale	0.010
1	SVG_linear	0.836	0.001	True	auto	0.010
2	SVG_linear	0.836	0.010	True	scale	0.010
3	SVG_linear	0.836	0.010	True	auto	0.010
4	SVG_linear	0.836	0.100	True	scale	0.010
5	SVG_linear	0.836	0.100	True	auto	0.010
6	SVG_linear	0.836	1.000	True	scale	0.010
7	SVG_linear	0.836	1.000	True	auto	0.010
8	SVG_linear	0.836	2.000	True	scale	0.010
9	SVG_linear	0.836	2.000	True	auto	0.010

Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión.



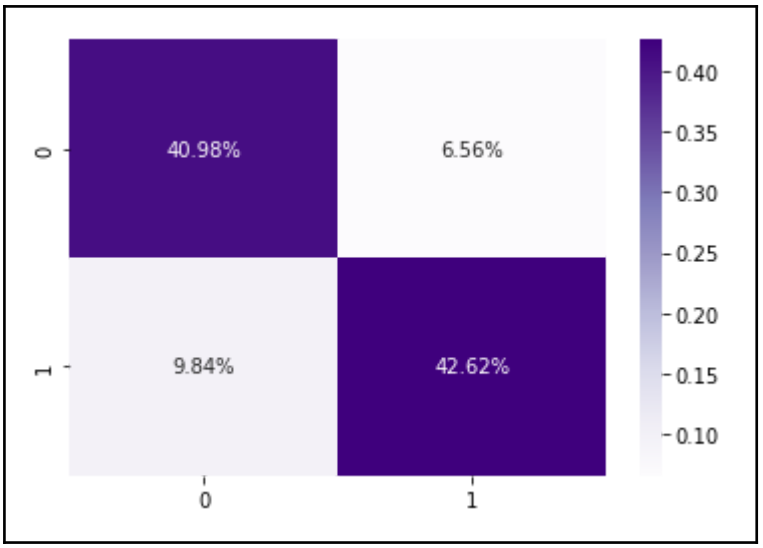
Resultados obtenidos para máquina de vectores de soporte con un kernel de tipo polinómico.

	Modelo	Precision	C	probability	gamma	tol
0	SVG_poly	0.836	0.001	True	scale	0.010
1	SVG_poly	0.836	0.001	True	auto	0.010
2	SVG_poly	0.836	0.010	True	scale	0.010
3	SVG_poly	0.836	0.010	True	auto	0.010
4	SVG_poly	0.836	0.100	True	scale	0.010
5	SVG_poly	0.836	0.100	True	auto	0.010
6	SVG_poly	0.836	1.000	True	scale	0.010
7	SVG_poly	0.836	1.000	True	auto	0.010
8	SVG_poly	0.836	2.000	True	scale	0.010
9	SVG_poly	0.836	2.000	True	auto	0.010

Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión:

UAB

Clasificación



Random Forest

Valores adoptados para el atributo estimadores

```
estimadores=[100,200,300,500]
```

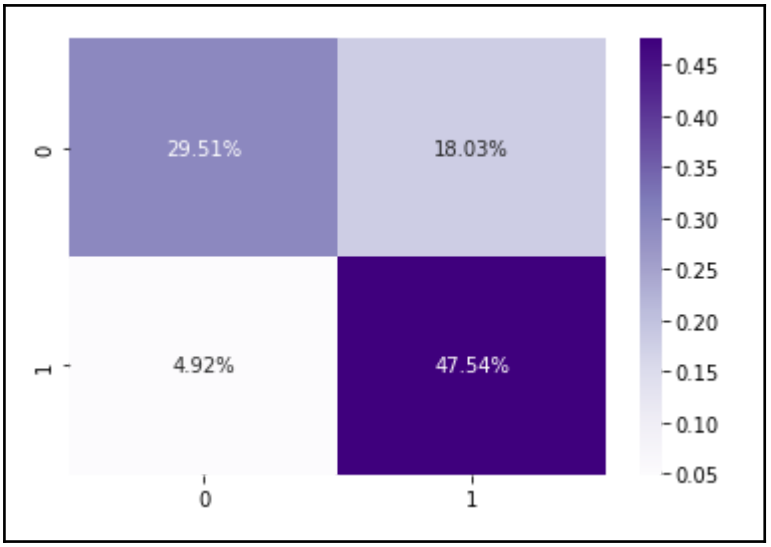
Valores de min_samples.

```
min_samples=[0.1,0.15,0.2,0.25,0.3]
```

Resultados obtenidos para Random Fores.

	Modelo	Precision	n_estimators	min_samples_leaf
1	Random Forest	0.885	100	0.150
0	Random Forest	0.869	100	0.100
16	Random Forest	0.869	500	0.150
15	Random Forest	0.869	500	0.100
5	Random Forest	0.869	200	0.100
12	Random Forest	0.869	300	0.200
17	Random Forest	0.852	500	0.200
11	Random Forest	0.852	300	0.150
10	Random Forest	0.852	300	0.100

Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión:



KNN

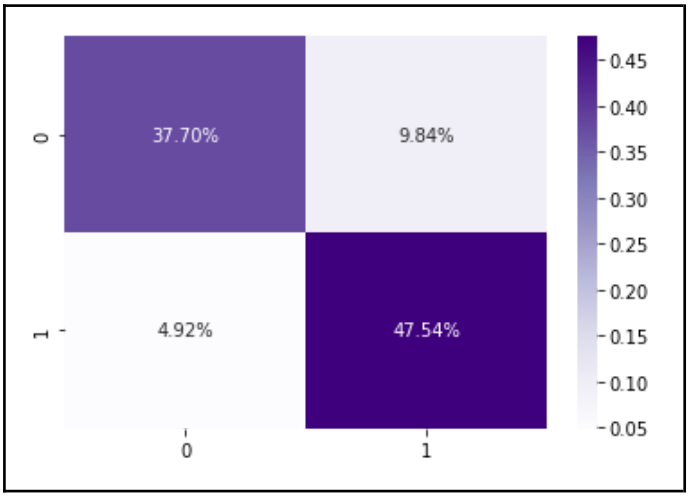
DIferentes valores probados para n

```
n=[10,15,20,30,50]
```

Resultados obtenidos para KNN

	Modelo	Precision	n
1	Knn	0.902	15
0	Knn	0.869	10
3	Knn	0.869	30
2	Knn	0.852	20
4	Knn	0.852	50

Matriz de confusión obtenida para la combinación de parámetros que nos dan la mayor precisión:



Resultado finales

	Modelo	Precision
4	Random Forest	0.902
5	Knn	0.902
1	SVG_rbf	0.885
0	Regresion Logistica	0.869
2	SVG_linear	0.869
3	SVG_poly	0.869

4. Crossvalidation

¿Por qué es importante cross-validar los resultados?

Ya que esto nos permite estimar la capacidad predictiva de los modelos cuando se aplican a nuevas observaciones, haciendo uso únicamente de los datos de entrenamiento.
El modelo se ajusta empleando un subconjunto de observaciones del conjunto de entrenamiento y se evalúa con las observaciones restantes.

¿Qué tan fiables serán los resultados obtenidos?

Esto dependerá en gran medida de la calidad de los datos es decir si el procesamiento de los datos nos hemos asegurado de que contamos con información clara, concisa y en la que no existan elementos nulos o elementos repetitivos que puedan dar lugar a predicciones erróneas la fiabilidad del modelo estará garantizada.

¿En qué casos será más fiable, si tenemos muchos datos de entrenamiento o pocos?

Cómo mencionaba anteriormente, no dependerá de la cantidad que tengamos sino más bien de cuán bien están tratados dichos datos, Sí que es verdad que en el caso de los datos de test, deberemos de reservar una cantidad modesta para poder generar resultados estadísticamente significativos.

¿Qué tipo de K-fold ha elegido? ¿Cuántos conjuntos ha seleccionado (cuál k)?

El número que hemos escogido es de 5.

¿Cómo afecta a los distintos valores de k?

En el método K-fold CV los k grupos empleados como entrenamiento son mucho menos solapantes, lo que se traduce en menor varianza al promediar las estimaciones de error, por lo cual la K que seleccionemos es determinante.

¿Es viable o conveniente aplicar `Leave One Out`?

LOO emplea n-1 observaciones para entrenar el modelo, lo que es prácticamente todo el set de datos disponible, maximizando así el ajuste del modelo a los datos disponibles y reduciendo el bias.

Lo ideal para hacer el Crossvalidation es tener sólo aquellos datos que sean qué carácter numérico en nuestro caso nuestro DataSet todos los datos son de carácter numérico.

Crossvalidation en base al RMSE

El número de Split seleccionado fue de 5.

```
Fold:1, Train set: 241, Test set:61
Fold:2, Train set: 241, Test set:61
Fold:3, Train set: 242, Test set:60
Fold:4, Train set: 242, Test set:60
Fold:5, Train set: 242, Test set:60
```

Resultados del random forest

```
Scores para cada fold son: [-0.1169377  -0.1387541  -0.13757541 -0.17608    -0.09817 ]
rmse= 0.37
```

Ajuste del modelo

```
For estimators: 50
rmse= 0.37
For estimators: 100
rmse= 0.37
For estimators: 150
rmse= 0.37
For estimators: 200
rmse= 0.37
For estimators: 250
rmse= 0.36
```

Resultados del Regresión Logística

```
Scores para cada fold son: [-0.16393443 -0.19672131 -0.14754098 -0.23333333 -0.13333333]
rmse= 0.42
```

Crossvalidation en base al accuracy

Resultados del random forest

```
Scores for each fold are: [0.85245902 0.80327869 0.81967213 0.75       0.85  ]
Average score: 0.82
```

Ajuste del modelo

```
Average score(50): 0.815
Average score(100): 0.815
Average score(150): 0.815
Average score(200): 0.818
Average score(250): 0.818
```

Average score(300): 0.828

Resultados de regresión Logística

Scores for each fold are: [0.83606557 0.80327869 0.85245902 0.76666667 0.86666667]
Average score: 0.83

Resultados de Leave One Out

Folds: 5, MSE: 0.6852459016393443

Folds: 5, MSE: 0.7621311475409837

5. Metric Analysis

¿Podría explicar y justificar cuál de las siguientes métricas será la más adecuada para su problema? `accuracy_score`, `f1_score` o `average_precision_score`.

En el caso del accuracy este mide cuantas observaciones tanto positivas como negativas se clasifican correctamente, debemos utilizar solo en problemas equilibrados.
El f1 combina la precisión junto con el recall, es decir si lo que nos importa es el recall a obtener más que la precisión, es decir si nos interesa más la relevancia.
Finalmente la average_precision_score combina tanto las presicion como el recall con lo cual tenemos todo en una misma visualización,y puede ofrecer una solución al dilema entre precision y recall de manera que podemos visualizar un punto medio entre ambos para encontrar un umbral que ofrezca el mejor modelo.

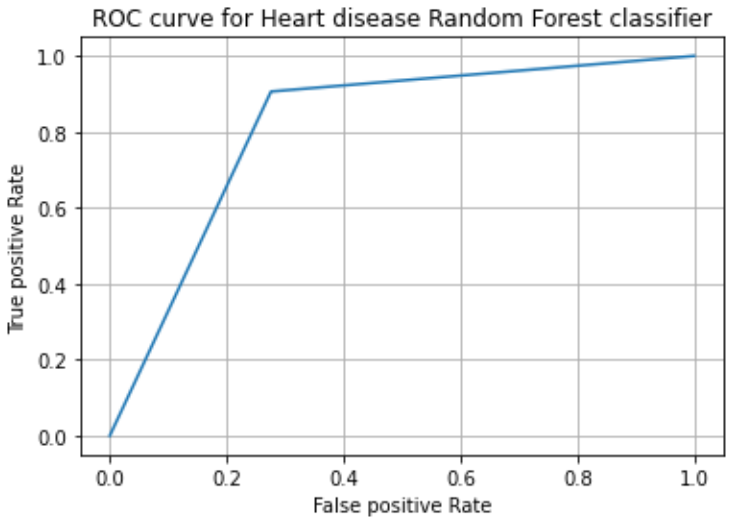
Evaluación del método seleccionado Random Forest

Acc : 82.00%
Precisión : 0.78
recall : 0.91
F1 score 0.84

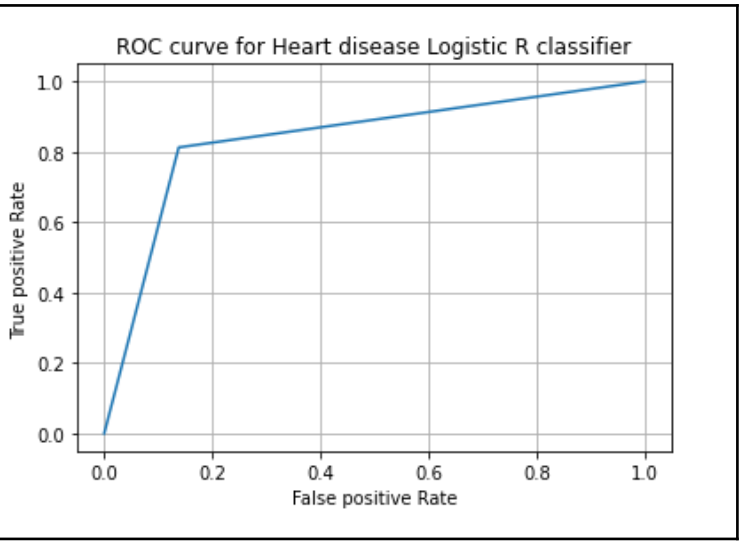
Evaluación del método con menor precisión

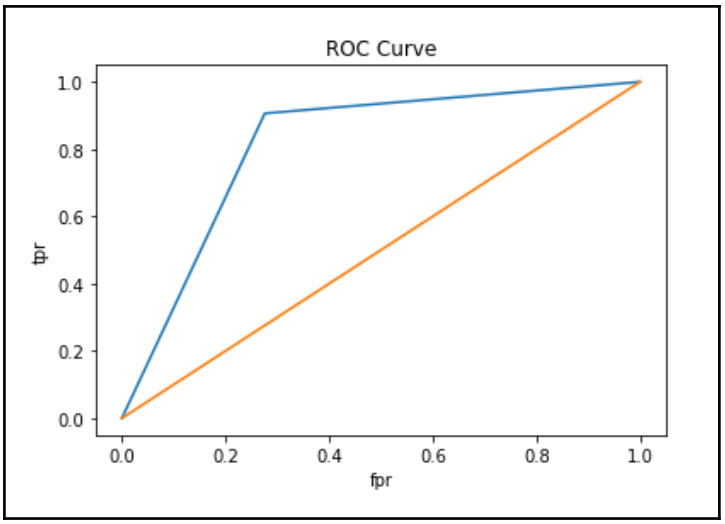
Acc : 84.00%
Precisión : 0.87
recall : 0.81
F1 score 0.84

ROC Curve para el Random Forest

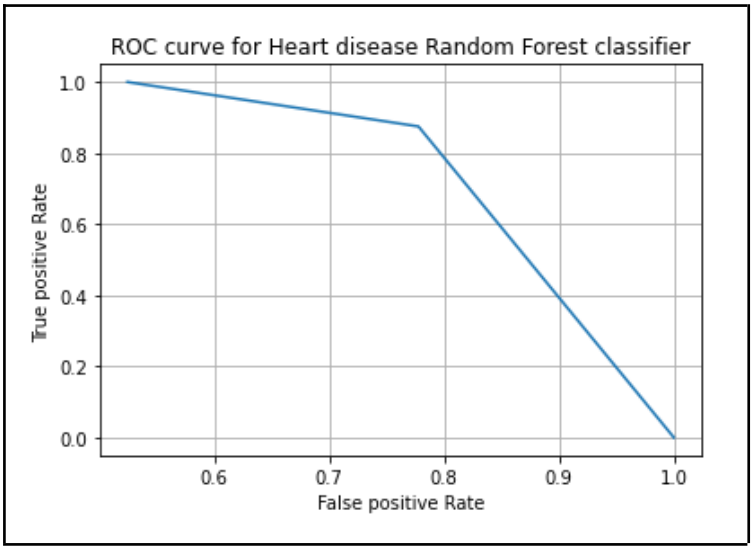


ROC Curve para Regresión Logística

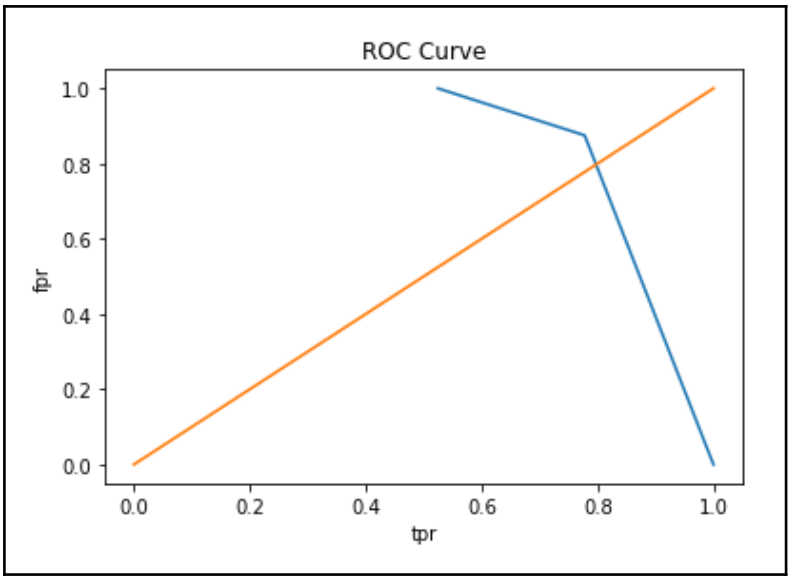




Precision Recall Curve



ROC Curve



Classification report

Resultado del Random Forest

	precision	recall	f1-score	support
Padece de enfermedad	0.84	0.72	0.78	29
No padece de enfermedad	0.78	0.88	0.82	32
accuracy			0.80	61
macro avg	0.81	0.80	0.80	61
weighted avg	0.81	0.80	0.80	61

Resultado del regresor logístico

	precision	recall	f1-score	support
Padece de enfermedad	0.81	0.86	0.83	29
No padece de enfermedad	0.87	0.81	0.84	32
accuracy			0.84	61
macro avg	0.84	0.84	0.84	61
weighted avg	0.84	0.84	0.84	61

6. Hyperparameter Search

¿Qué formas de buscar el mejor parámetro ha encontrado? ¿Son costosas computacionalmente hablando?

En este caso para poder encontrar los parámetros que nos diese un mejor resultado hemos realizado las diferentes operaciones es decir calculado el modelo varias veces con diferentes parámetros de entrada, lo cual a la larga computacionalmente hablando son muy costosas.

¿Existen otros métodos de búsqueda más eficientes?

Sí que existen y de hecho podemos encontrar varias librerías que nos ofrecen diferentes funciones que nos permiten encontrar aquellos parámetros que sean los más óptimos para nuestro modelo .

Encontrando parámetros que mejor se ajusten a nuestro modelo.

```
tuned_parameters = {'min_samples_leaf': range(10,100,10), 'n_estimators'
: range(10,100,10),'max_features':['auto','sqrt','log2']}
```

```
RR_model= RandomizedSearchCV(model_RR,
tuned_parameters,cv=10,scoring='accuracy',n_iter=20,n_jobs= -1)
```

```
print(RR_model.best_params_)
```

El resultado obtenido fue el siguiente:

```
{'n_estimators': 50, 'min_samples_leaf': 20, 'max_features': 'sqrt'}
```