Aprendizaje de máquinas

Default of credit card clients
Breast cancer wisconsin

Ramses Alexander Coraspe Valdez

Default of credit card clients

DATASET: https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

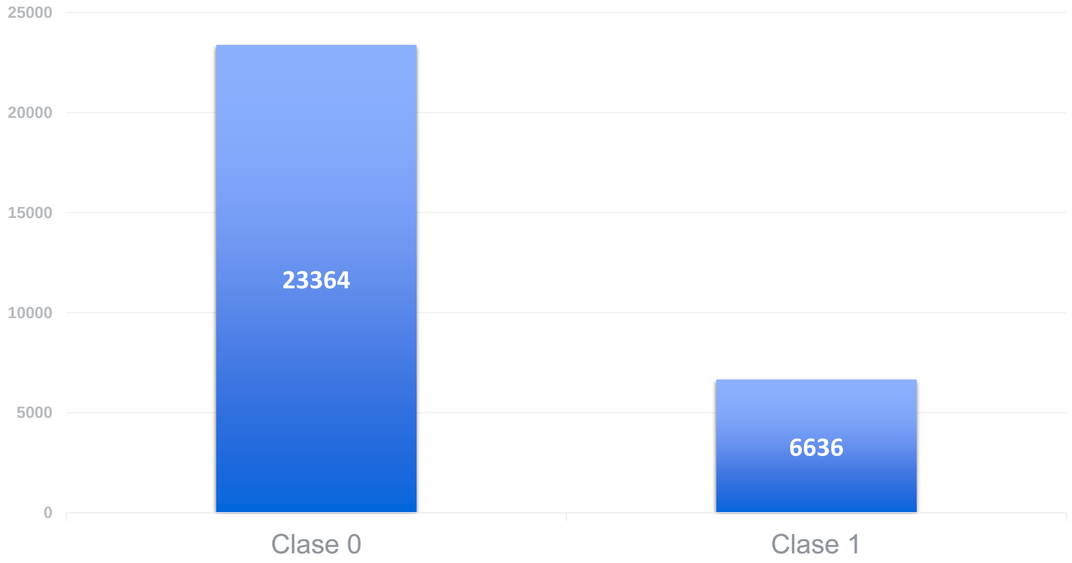
OBJETIVO: Comparar las precisiones predictivas de la probabilidad de incumplimiento de los pagos de tarjetas de crédito usando 5 algoritmos de aprendizaje de maquinas (**Logistic Regression, ANN, KNN, SVM, Naive Bayes**)

VARIABLES:

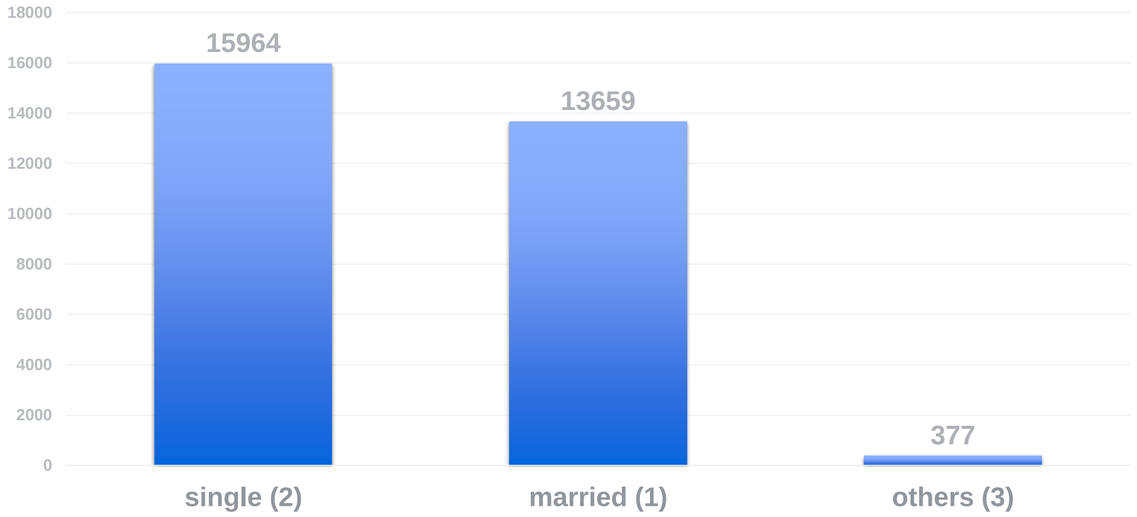
Cantidad de registros: 30000
Numéricas: LIMIT_BAL, AGE, BILL_AMT1, PAY_AMT1, PAY_AMT2, PAY_AMT3, PAY_AMT4, PAY_AMT5, PAY_AMT6
Categóricas: SEX, EDUCATION, MARRIAGE, PAY_2, PAY_3, PAY_4, PAY_5, PAY_6

La variable "**Default_payment_next_month**" es dependiente y representa a la clase, 1= SI, 0=NO.

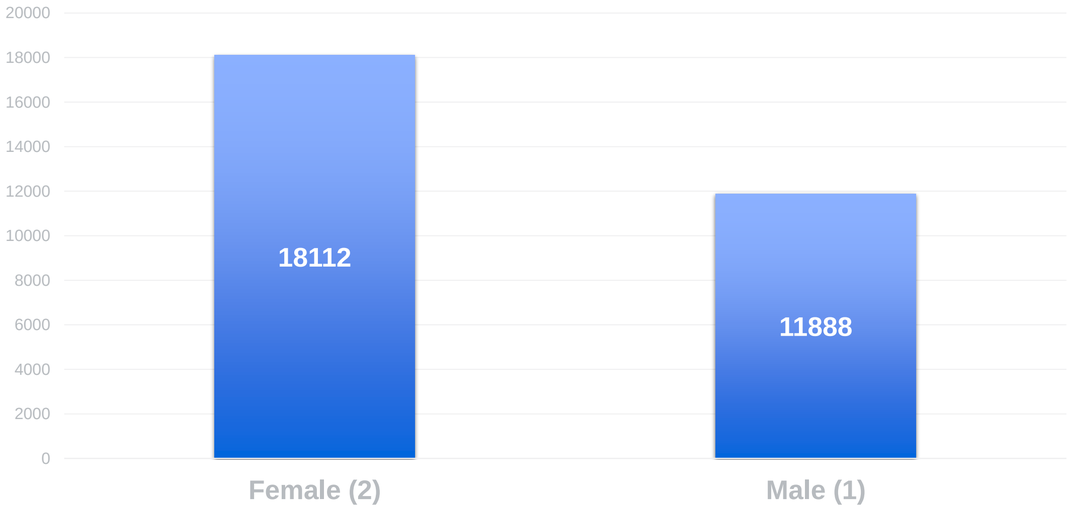
Default_payment_next_month



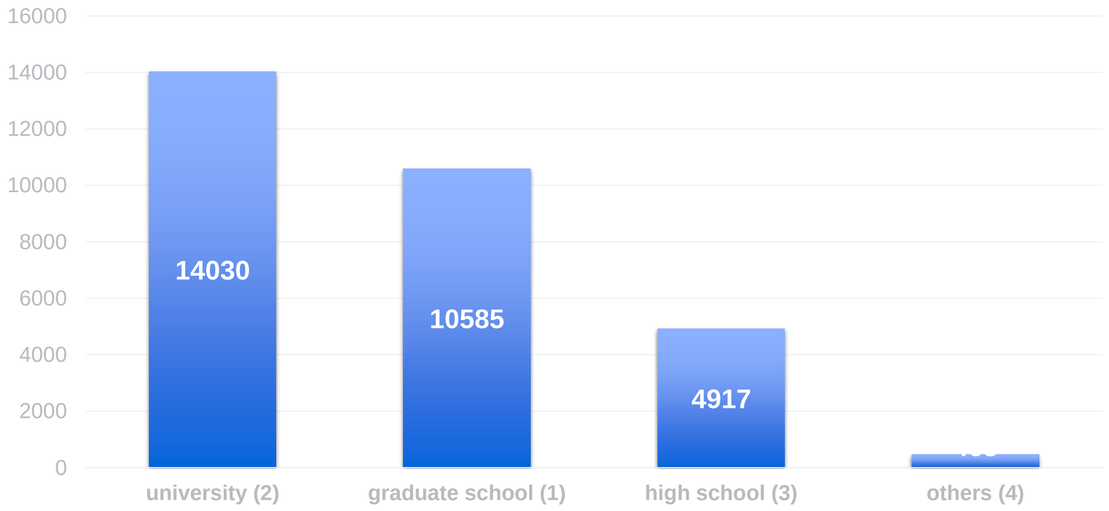
Marriage

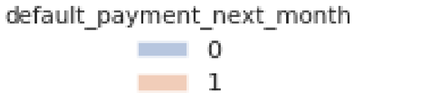


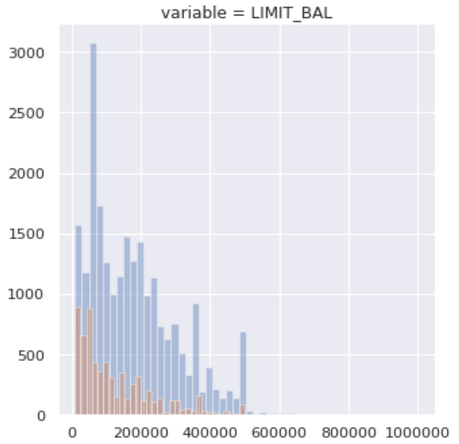
Sex



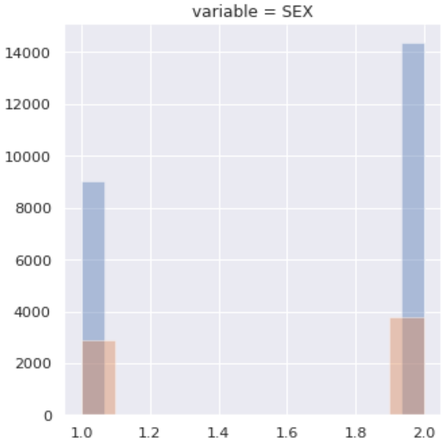
Education



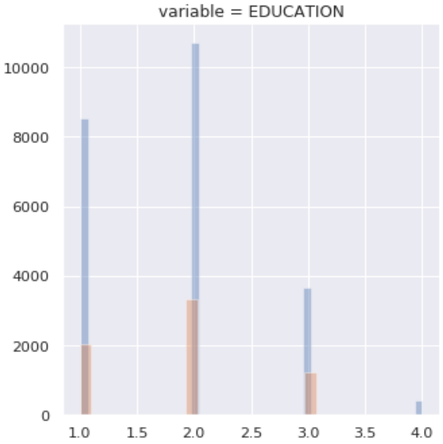




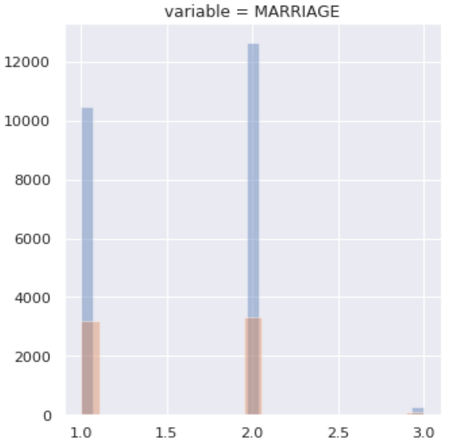
Los incumplimientos tienen una mayor proporción de valores LIMIT_BAL bajos



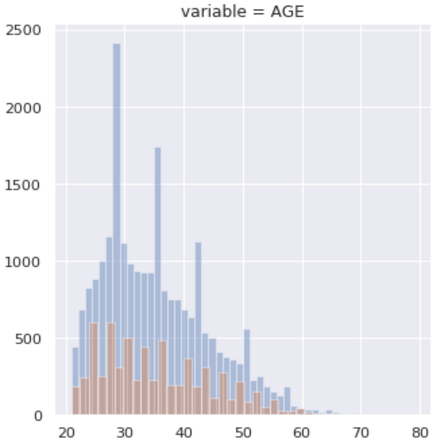
Las mujeres cumplen mas.



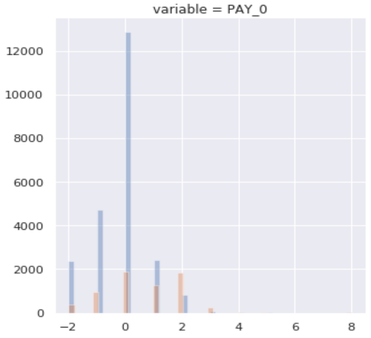
Los clientes con grados de educación mas altos cumplen mas

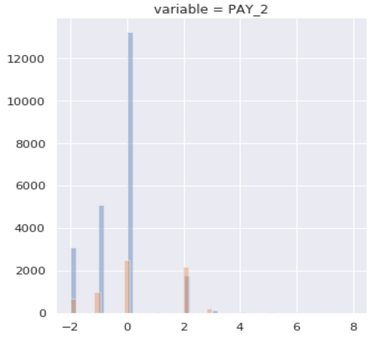


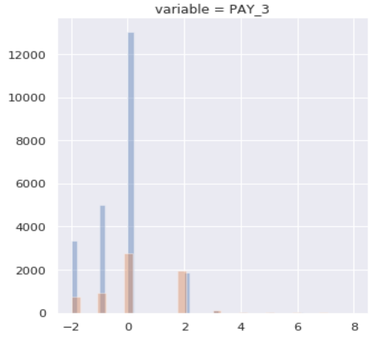
Hay una ligera diferencia, las personas solteras cumplen mas.

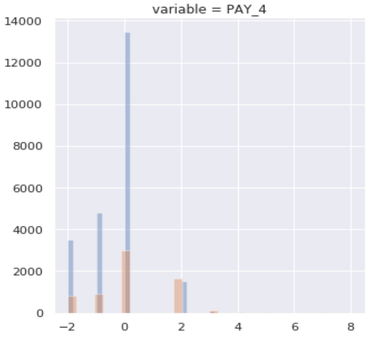


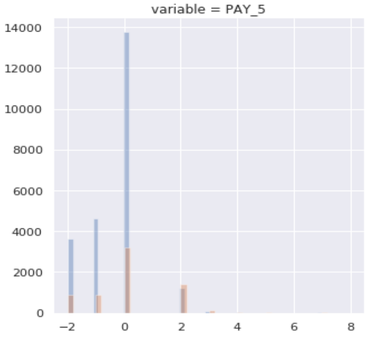
Los clientes con un rango de edad entre 28 y 40 años cumplen mas.

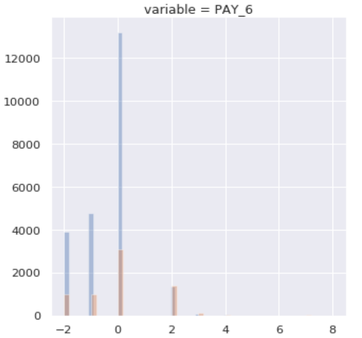




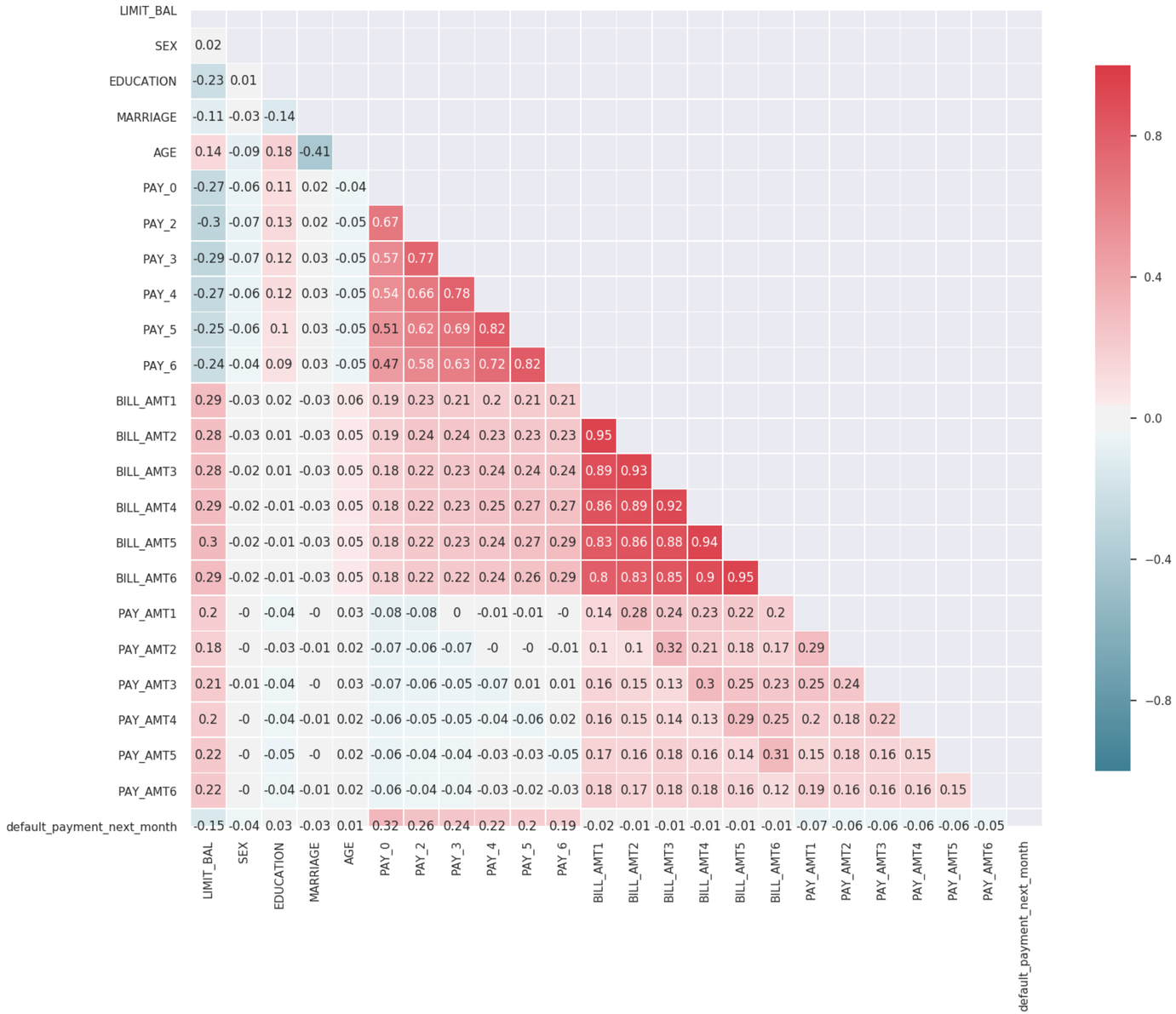








Las variables PAY_N son importantes, los cumplimientos del pago tienen una mayor proporción de ceros o negativos, adelantarse al pago o estar al día esta relacionado a no incumplir el mes siguiente.



BILL_AMT2 Altamente correlacionada con BILL_AMT1

BILL_AMT3 Altamente correlacionada con BILL_AMT2

BILL_AMT4 Altamente correlacionada con BILL_AMT3

BILL_AMT5 Altamente correlacionada con BILL_AMT4

BILL_AMT6 Altamente correlacionada con BILL_AMT5

Resumen del análisis y decisiones

1. La clase 1 representa el 22.12 % de los registros y la clase 0 representa el 77.88%

2. No hay valores faltantes

3. El dataset tiene **35 registros duplicados**, no fueron eliminados por las siguientes razones:

a. No tengo mas información de la naturaleza del conjunto de datos, no se si es un error en la carga de los datos.

b. Las repeticiones son las que proporcionan el peso de la evidencia.

c. El propósito del conjunto de entrenamiento es convertir ese grupo de datos en información, acumular experiencia de la vida real, lo que no lograremos si perdemos información de su frecuencia.

4. Las variables BILL_AMT2, BILL_AMT3, BILL_AMT4, BILL_AMT5, BILL_AMT6 fueron eliminadas ya que tienen la misma información que la variable BILL_AMT1.

5. Las variables "MARRIAGE", "SEX" , "EDUCATION" tenían más clases de lo que la documentación indicaba, se tomo la decisión de asignar esos registros a categorías ya existentes.

6. Las variables independientes fueron escaladas usando la librería StandardScaler

7. Los experimentos para el entrenamiento del modelo fueron hechos con **Python** y no usaron variables ficticias, debido a que no mejoraba la precisión en los resultados.

Comparación de la precisión de los algoritmos

La tabla siguiente muestra los resultados de cada algoritmo para cada partición, el mejor desempeño de cada algoritmo esta resaltados en color verde y el peor desempeño esta resaltado en color rojo la decisión fue tomada basándome en el valor **F1_score** y el **AUC,** el algoritmo que mejor presenta desempeño es el **Perceptrón multicapa** con la partición del 30% para test, debido a que tiene un valor de F1_score mas elevado y un valor AUC bueno.

Algoritmo	Training (90%)	Test (10%)	Training (80%)	Test (20%)	Training (70%)	Test (30%)
Regresión logística	**Accuracy: 0.8120** **F1_score: 0.3591** **AUC: 0.6065**	**Accuracy: 0.7976** **F1_score: 0.3380** **AUC: 0.5964**	Accuracy: 0.8102 F1_score: 0.3512 AUC: 0.6033	Accuracy: 0.8108 F1_score: **0.3620** AUC: 0.6072	Accuracy: 0.8101 F1_score: 0.3501 AUC: 0.6029	Accuracy: 0.8111 F1_score: 0.3594 AUC: 0.6064
Perceptrón Multicapa	Accuracy: 0.8284 F1_score: 0.4882 AUC: 0.6644	Accuracy: 0.8153 F1_score: 0.4733 AUC: 0.6563	Accuracy: 0.8281 F1_score: 0.4939 AUC: 0.6673	Accuracy: 0.8238 F1_score: 0.4851 AUC: 0.6636	**Accuracy: 0.8317** **F1_score: 0.5152** **AUC: 0.6785**	**Accuracy: 0.8232** **F1_score: 0.4976** **AUC: 0.6705**
KNN	Accuracy: 0.8216 F1_score: 0.4689 AUC: 0.6550	Accuracy: 0.814 F1_score: 0.4725 AUC: 0.6559	Accuracy: 0.8207 F1_score: 0.4655 AUC: 0.6532	Accuracy: 0.8216 F1_score: **0.4785** AUC: 0.6603	Accuracy: 0.8186 F1_score: 0.4531 AUC: 0.6471	Accuracy: 0.8182 F1_score: 0.4639 AUC: 0.6528
SVM	Accuracy: 0.8255 F1_score: 0.4680 AUC: 0.6543	Accuracy: 0.818 F1_score: 0.4657 AUC: 0.6523	Accuracy: 0.8237 F1_score: 0.4650 AUC: 0.6527	Accuracy: 0.8253 F1_score: **0.4749** AUC: 0.6580	Accuracy: 0.823 F1_score: 0.4604 AUC: 0.6505	Accuracy: 0.824 F1_score: 0.4680 AUC: 0.6545
Naive Bayes	Accuracy: 0.7181 F1_score: 0.5014 AUC: 0.6911	Accuracy: 0.7263 F1_score: **0.5196** AUC: 0.6991	Accuracy: 0.7212 F1_score: 0.5036 AUC: 0.6915	Accuracy: 0.7256 F1_score: 0.5098 AUC: 0.6984	Accuracy: 0.7248 F1_score: 0.5033 AUC: 0.6906	Accuracy: 0.73 F1_score: 0.5138 AUC: 0.7006

Configuración de los algoritmos:

Regresión logística: lg = linear_model.LogisticRegression(random_state = 40, max_iter = 500,solver='lbfgs’)

Perceptron multicapa: classifier = MLPClassifier(hidden_layer_sizes=(120, 80, 40, 10), max_iter=2000, activation = 'relu', alpha= 0.5, solver='sgd', random_state=40)

KNN: knn = KNeighborsClassifier(n_neighbors=23)

SVM: svclassifier = SVC(kernel='rbf’)

Matrices de confusión

Algoritmo	Training (90%)	Test (10%)	Training (80%)	Test (20%)	Training (70%)	Test (30%)
Regresión logística	**Accuracy: 0.8120** **F1_score: 0.3591** **AUC: 0.6065**	**Accuracy: 0.7976** **F1_score: 0.3380** **AUC: 0.5964**	Accuracy: 0.8102 F1_score: 0.3512 AUC: 0.6033	Accuracy: 0.8108 F1_score: **0.3620** AUC: 0.6072	Accuracy: 0.8101 F1_score: 0.3501 AUC: 0.6029	Accuracy: 0.8111 F1_score: 0.3594 AUC: 0.6064
Perceptrón Multicapa	Accuracy: 0.8284 F1_score: 0.4882 AUC: 0.6644	Accuracy: 0.8153 F1_score: 0.4733 AUC: 0.6563	Accuracy: 0.8281 F1_score: 0.4939 AUC: 0.6673	Accuracy: 0.8238 F1_score: 0.4851 AUC: 0.6636	**Accuracy: 0.8317** **F1_score: 0.5152** **AUC: 0.6785**	**Accuracy: 0.8232** **F1_score: 0.4976** **AUC: 0.6705**

TP= 2238	FP=78
FN=529	TN=155

TP= 6621	FP=398
FN=1193	TN=788

Breast cancer wisconsin

DATASET: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

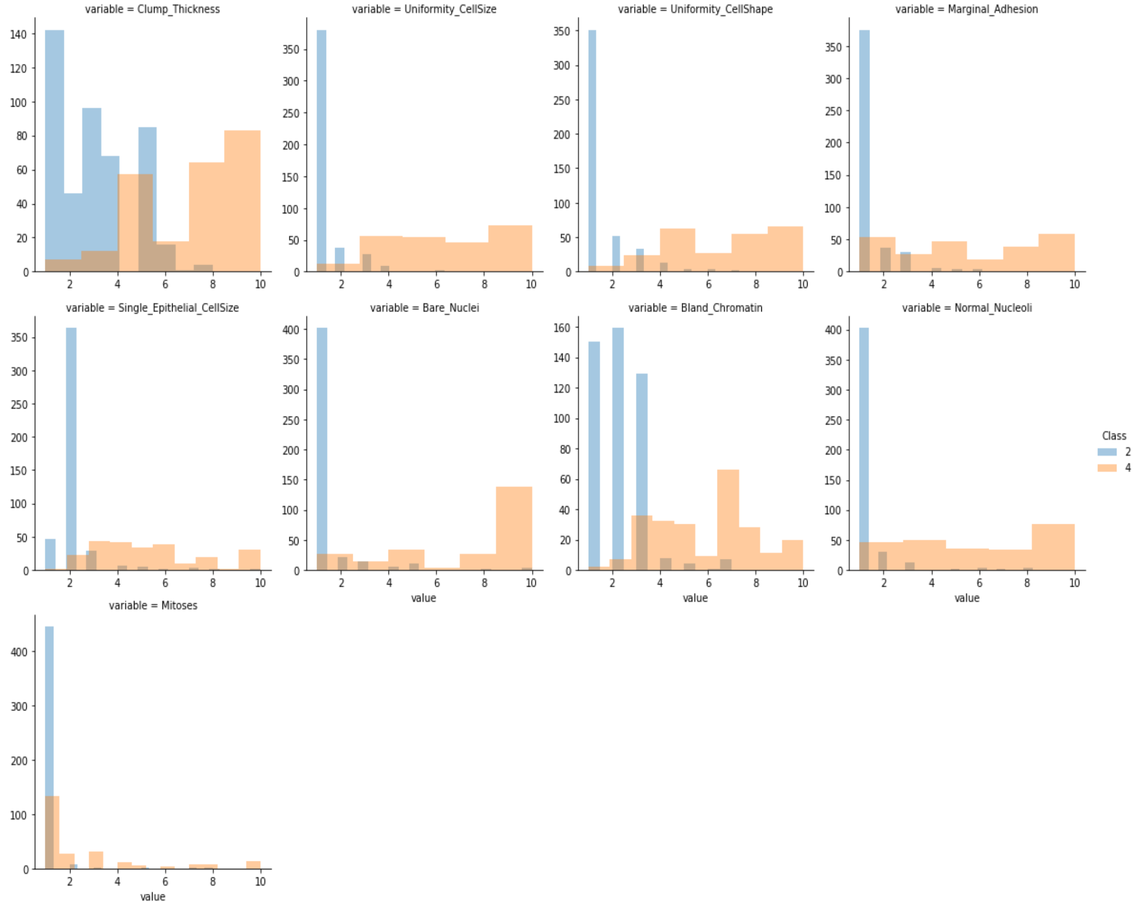
OBJETIVO: Comparar las precisiones predictivas de la probabilidad de tumores malignos a partir de los datos del núcleo de una célula. usando 5 algoritmos de aprendizaje de maquinas (**Logistic Regression, ANN, KNN, SVM, Naive Bayes**)

VARIABLES:

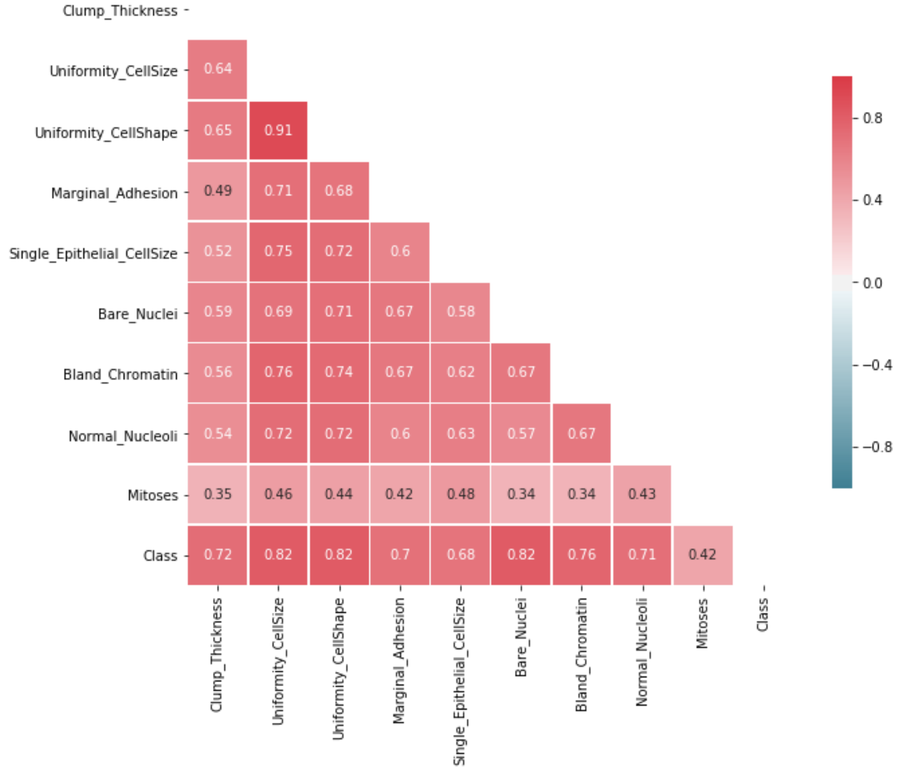
Cantidad de registros: 699

Las primeras nueve variables tienen valores nominales y también se les asignaron valores ordinales, con la misma distancia entre categorías. Estas variables podrían normalizarse a otro rango, pero en este caso todas tienen el mismo comportamiento, aparentemente ya están escaladas. Estas variables serán las variables de entrada.

La variable "Clase" es categórica (binaria) y se utilizará para identificar la clase del registro. 2= Benigno, 4=Maligno.



"'clump thickness" se distribuye uniformemente en cierta medida. Todas las demás variables están sesgadas a la derecha.



Al verificar la matriz de correlación, nos damos cuenta de que las variables Uniformity_CellSize y Uniformity_CellShape están altamente correlacionadas y cualquiera de ellas podría ser rechazada para el análisis.

Resumen del análisis y decisiones

1. La clase “Benigno" representa el 65.52% y la clase "Maligno" representa el 34.48%

2. Hay 16 valores faltantes en la columna **Bare_Nuclei**, fueron sustituidos con el valor medio del conjunto de valores, no se desvía demasiado de los datos proporcionados.

3. El dataset no tiene registros duplicados.

4. La variables *Uniformity_CellShape* fue eliminada ya que tiene la misma información que la variable *Uniformity_CellSize.*

7. Los experimentos para el entrenamiento del modelo fueron hechos con **Python** y no usaron variables ficticias, debido a que no mejoraba la precisión en los resultados.

Comparación de la precisión de los algoritmos

La tabla siguiente muestra los resultados de cada algoritmo para cada partición, el mejor desempeño de cada algoritmo esta resaltado en color verde y el peor desempeño esta resaltado en color rojo la decisión fue tomada basándome en el valor **F1_score** y el **AUC,** el algoritmo que mejor presenta desempeño es el **KNN** con la partición del 30% para test, debido a que tiene un valor de F1_score mas elevado y un valor AUC considerablemente bueno.

Algoritmo	Training (90%)	Test (10%)	Training (80%)	Test (20%)	Training (70%)	Test (30%)
Regresión logística	Accuracy: 0.9729 F1_score: 0.9607 AUC: 0.9685	Accuracy: 0.9285 F1_score: 0.8979 AUC: 0.9357	Accuracy: 0.9749 F1_score: 0.9633 AUC: 0.9721	Accuracy: 0.9357 F1_score: 0.9090 AUC: 0.9277	Accuracy: 0.9754 F1_score: 0.9651 AUC: 0.9730	Accuracy: 0.9476 F1_score: 0.9197 AUC: 0.9387
Perceptrón Multicapa	**Accuracy: 0.9984** **F1_score: 0.9977** **AUC: 0.9977**	**Accuracy: 0.9142** **F1_score: 0.8749** **AUC: 0.9139**	Accuracy: 1 F1_score: 1 AUC: 1	Accuracy: 0.95 F1_score: 0.9320 AUC: 0.9522	Accuracy: 1 F1_score: 1 AUC: 1	Accuracy: 0.9380 F1_score: 0.9064 AUC: 0.9316
KNN	Accuracy: 0.9761 F1_score: 0.9655 AUC: 0.9731	Accuracy: 0.9285 F1_score: 0.8979 AUC: 0.9357	Accuracy: 0.9785 F1_score: 0.9685 AUC: 0.9761	Accuracy: 0.95 F1_score: 0.9292 AUC: 0.9433	**Accuracy: 0.9815** **F1_score: 0.9739** **AUC: 0.9804**	**Accuracy: 0.9571** **F1_score: 0.9343** **AUC: 0.9495**
SVM	Accuracy: 0.9793 F1_score: 0.9702 AUC: 0.9777	Accuracy: 0.9285 F1_score: 0.8979 AUC: 0.9357	Accuracy: 0.9803 F1_score: 0.9715 AUC: 0.9812	Accuracy: 0.9357 F1_score: 0.9108 AUC: 0.9322	Accuracy: 0.9836 F1_score: 0.9770 AUC: 0.9847	Accuracy: 0.9428 F1_score: 0.9154 AUC: 0.9426
Naive Bayes	Accuracy: 0.9634 F1_score: 0.9487 AUC: 0.9666	Accuracy: 0.9428 F1_score: 0.92 AUC: 0.9574	Accuracy: 0.9677 F1_score: 0.9543 AUC: 0.9717	Accuracy: 0.9357 F1_score: 0.9142 AUC: 0.9411	Accuracy: 0.9693 F1_score: 0.9577 AUC: 0.9736	Accuracy: 0.9428 F1_score: 0.9166 AUC: 0.9463

Configuración de los algoritmos:

Regresión logística: lg = linear_model.LogisticRegression(random_state = 40, max_iter = 100,solver='lbfgs')
Perceptron multicapa: clf = MLPClassifier(solver='lbfgs', max_iter=2000, random_state=40, activation='relu', alpha= 0.001, hidden_layer_sizes=(10,5 ,2))
KNN: knn = KNeighborsClassifier(n_neighbors=7)
SVM: svclassifier = SVC(kernel='rbf')

Matrices de confusión

Algoritmo	Training (90%)	Test (10%)	Training (80%)	Test (20%)	Training (70%)	Test (30%)
Perceptrón Multicapa	**Accuracy: 0.9984 F1_score: 0.9977 AUC: 0.9977**	**Accuracy: 0.9142 F1_score: 0.8749 AUC: 0.9139**	Accuracy: 1 F1_score: 1 AUC: 1	Accuracy: 0.95 F1_score: 0.9320 AUC: 0.9522	Accuracy: 1 F1_score: 1 AUC: 1	Accuracy: 0.9380 F1_score: 0.9064 AUC: 0.9316
KNN	Accuracy: 0.9761 F1_score: 0.9655 AUC: 0.9731	Accuracy: 0.9285 F1_score: 0.8979 AUC: 0.9357	Accuracy: 0.9785 F1_score: 0.9685 AUC: 0.9761	Accuracy: 0.95 F1_score: 0.9292 AUC: 0.9433	**Accuracy: 0.9815 F1_score: 0.9739 AUC: 0.9804**	**Accuracy: 0.9571 F1_score: 0.9343 AUC: 0.9495**

TP= 43	FP=4
FN=2	TN=21

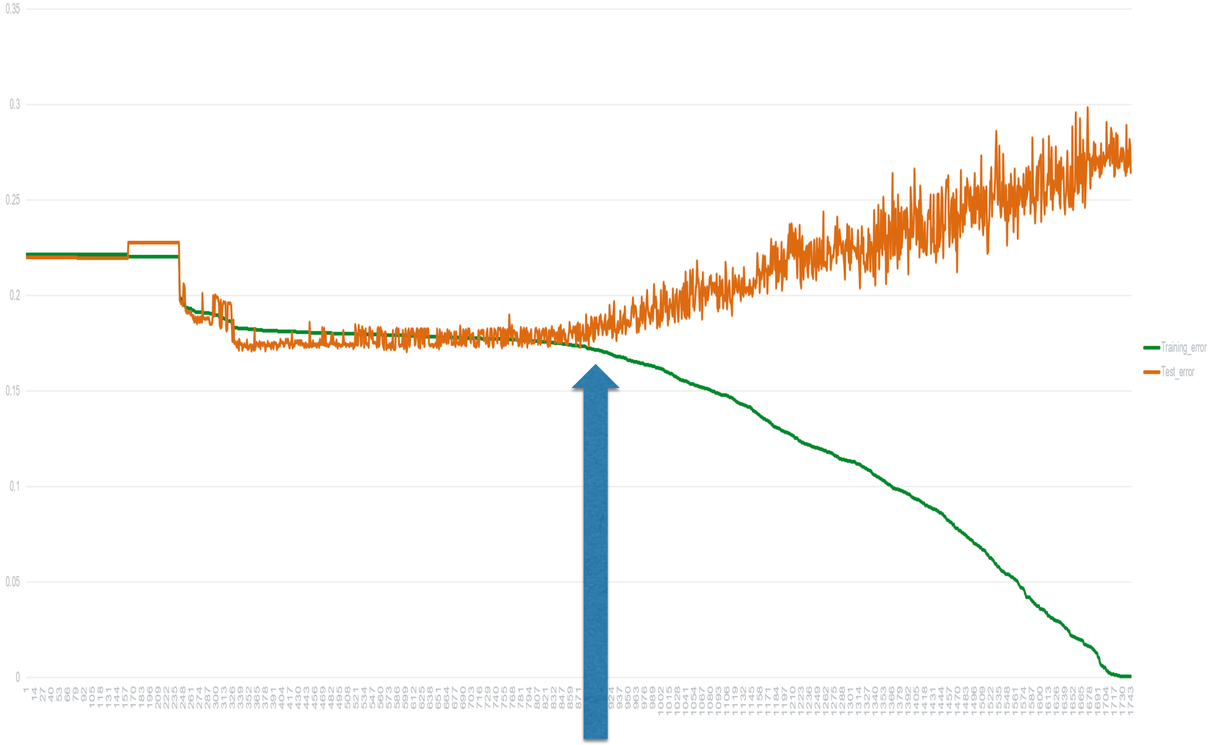
TP= 137	FP=4
FN=5	TN=64

Conclusiones

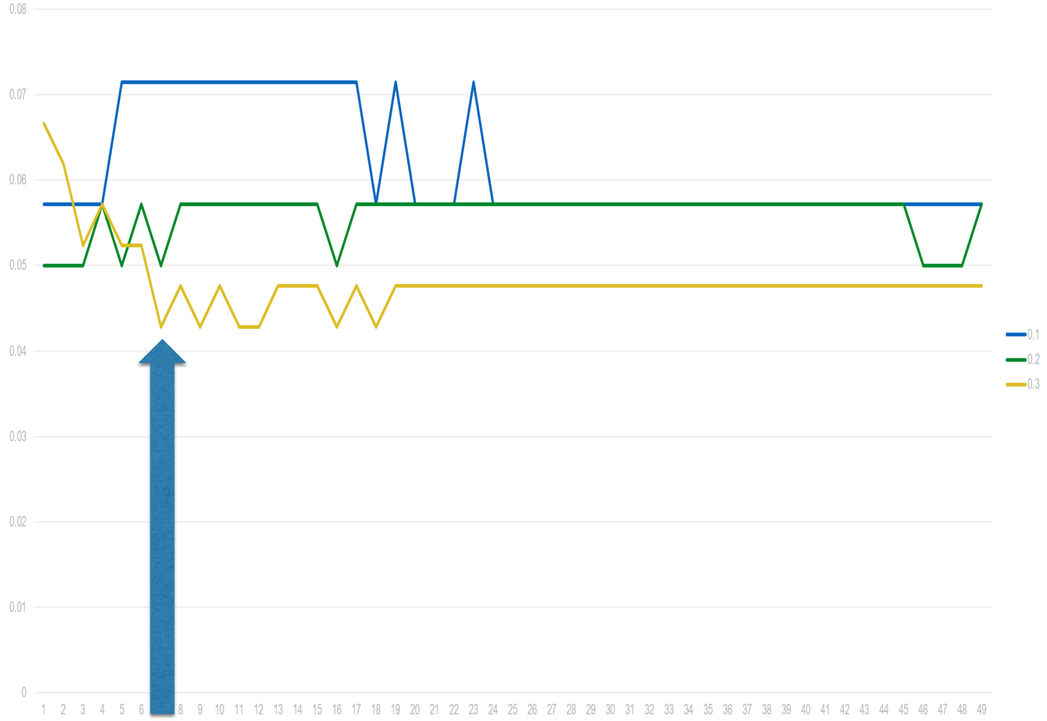
El método del **perceptrón multicapa** ofrece mejores resultados para el primer conjunto de datos usando una partición para test del 30%, aunque tiene mas complejidad en el tiempo, la elección de los hiperparametros se basó usando la grafica mostrada abajo, se tomaron aleatoriamente varias configuraciones de función de activación, optimizador, taza de aprendizaje y configuraciones de capas ocultas, y la configuración seleccionada se basa en una que mantuviera el error de Training muy cercano al error de Test, antes de ver la divergencia entre ambos errores. **Grafica 1.**

El método del **KNN** ofrece mejores resultados para el segundo conjunto de datos, usando una partición para test del 30%, la búsqueda del valor K se basó usando la grafica mostrada abajo, se seleccionó un valor de K que tuviera el error mas bajo. **Grafica 2.**

Código:
https://wittline.github.io/MachineLearning/Algorithm%20comparison/Pages/Breast_cancer_wisconsin.html
https://wittline.github.io/MachineLearning/Algorithm%20comparison/Pages/Default_of_credit_card_clients.html



Grafica 1



Grafica 2