Tecnológico de Monterrey

Proyecto Final Aprendizaje Automático

José Juan Suárez Ramos
A01224078

Kickstarted Database

Problemática

Mediante la base de datos de kickstarted, pronosticar si un proyecto podrá obtener el monto objetivo en dólares (usd_goal_real) a través de 5 modelos de clasificación.

Primeros Pasos

Analizando los datos

- Tipos de Datos, categóricos, fechas y numéricos

```
data.head()
```

	ID	name	category	main_category	currency	deadline	goal	launched	pledged	state	backers	country	usd pledged	usd_pledged_real	usd_goal_real
0	1000002330	The Songs of Adelaide & Abullah	Poetry	Publishing	GBP	2015-10-09	1000.0	2015-08-11 12:12:28	0.0	failed	0	GB	0.0	0.0	1533.95
1	1000003930	Greeting From Earth: ZGAC Arts Capsule For ET	Narrative Film	Film & Video	USD	2017-11-01	30000.0	2017-09-02 04:43:57	2421.0	failed	15	US	100.0	2421.0	30000.00
2	1000004038	Where is Hank?	Narrative Film	Film & Video	USD	2013-02-26	45000.0	2013-01-12 00:20:50	220.0	failed	3	US	220.0	220.0	45000.00
3	1000007540	ToshiCapital Rekordz Needs Help to Complete Album	Music	Music	USD	2012-04-16	5000.0	2012-03-17 03:24:11	1.0	failed	1	US	1.0	1.0	5000.00
4	1000011046	Community Film Project: The Art of Neighborhoo...	Film & Video	Film & Video	USD	2015-08-29	19500.0	2015-07-04 08:35:03	1283.0	canceled	14	US	1283.0	1283.0	19500.00

```
ID                  int64
name               object
category           object
main_category      object
currency           object
deadline           object
goal              float64
launched           object
pledged           float64
state              object
backers             int64
country            object
usd pledged       float64
usd_pledged_real  float64
usd_goal_real     float64
dtype: object
```

- Data Types => Como nuestro problema es de clasificación, debemos de considerar transformar los datos categóricos y fechas a valores numéricos para que nuestros modelos sean más efectivos.

Nueva variable de salida

Como queremos calcular si un proyecto llegó a su objetivo o no, creamos un nuevo atributo llamado goal_reached

goal_reached (bool) => será nuestra nueva variable de salida, se calculará considerando que **Usd Pledged Real** sea mayor o igual a **Usd Goal Real**, con eso tendremos valores 1s y 0s.

```
# Transformación de Datos

data["name"] = encoder.fit_transform(data['name'].astype(str))
data["main_category"] = encoder.fit_transform(data["main_category"].astype(str))
data["currency"] = encoder.fit_transform(data["currency"].astype(str))
data["state"] = encoder.fit_transform(data["state"].astype(str))
data["category"] = encoder.fit_transform(data["category"].astype(str))
data["country"] = encoder.fit_transform(data["country"].astype(str))
data["goal"] = minmax_scaling(data["goal"], columns = [0])
data['deadline'] = pd.to_datetime(data['deadline'])
data["goal_reached"] = data["usd_pledged_real"] >= data["usd_goal_real"] # Nuestra nueva variable de salida
data["usd_goal_real"] = stats.boxcox(data["usd_goal_real"])[0]
data["goal_reached"] = encoder.fit_transform(data["goal_reached"].astype(bool))
data['deadline']= data['deadline'].map(dt.datetime.toordinal)
data['launched'] = pd.to_datetime(data['launched'])
data['launched']= data['launched'].map(dt.datetime.toordinal)
```

Llevamos a cabo una normalización de “usd_goal_real” para poder eliminar de una forma más sencilla los outliers

Resultado de transformación de Datos

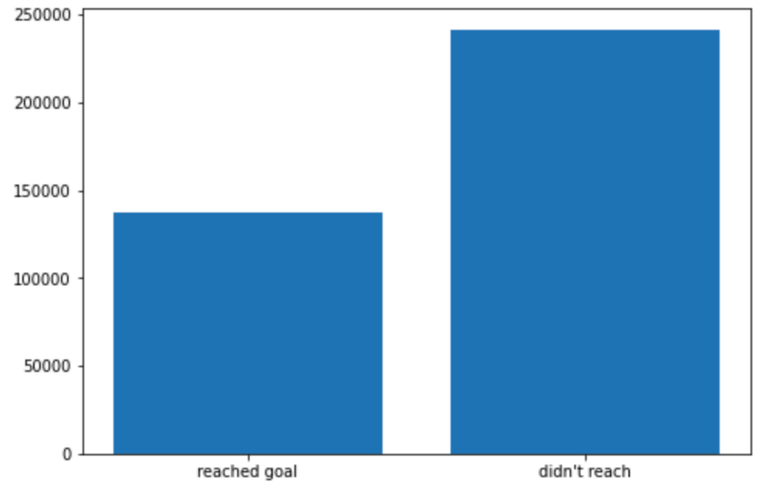
Nueva tabla con puros datos numéricos.

```
data.head()
```

	ID	name	category	main_category	currency	deadline	goal	launched	pledged	state	backers	country	usd pledged	usd_pledged_real	usd_goal_real	goal_reached
0	1000002330	326290	108	12	5	735880	0.000010	735821	0.0	1	0	9	0.0	0.0	7.977321	0
1	1000003930	132984	93	6	13	736634	0.000300	736574	2421.0	1	15	22	100.0	2421.0	11.605945	0
2	1000004038	357882	93	6	13	734925	0.000450	734880	220.0	1	3	22	220.0	220.0	12.119890	0
3	1000007540	338194	90	10	13	734609	0.000050	734579	1.0	1	1	22	1.0	1.0	9.390248	0
4	1000011046	76586	55	6	13	735839	0.000195	735783	1283.0	0	14	22	1283.0	1283.0	11.065033	0

Resultado de nueva variable de salida

```
<BarContainer object of 2 artists>
```



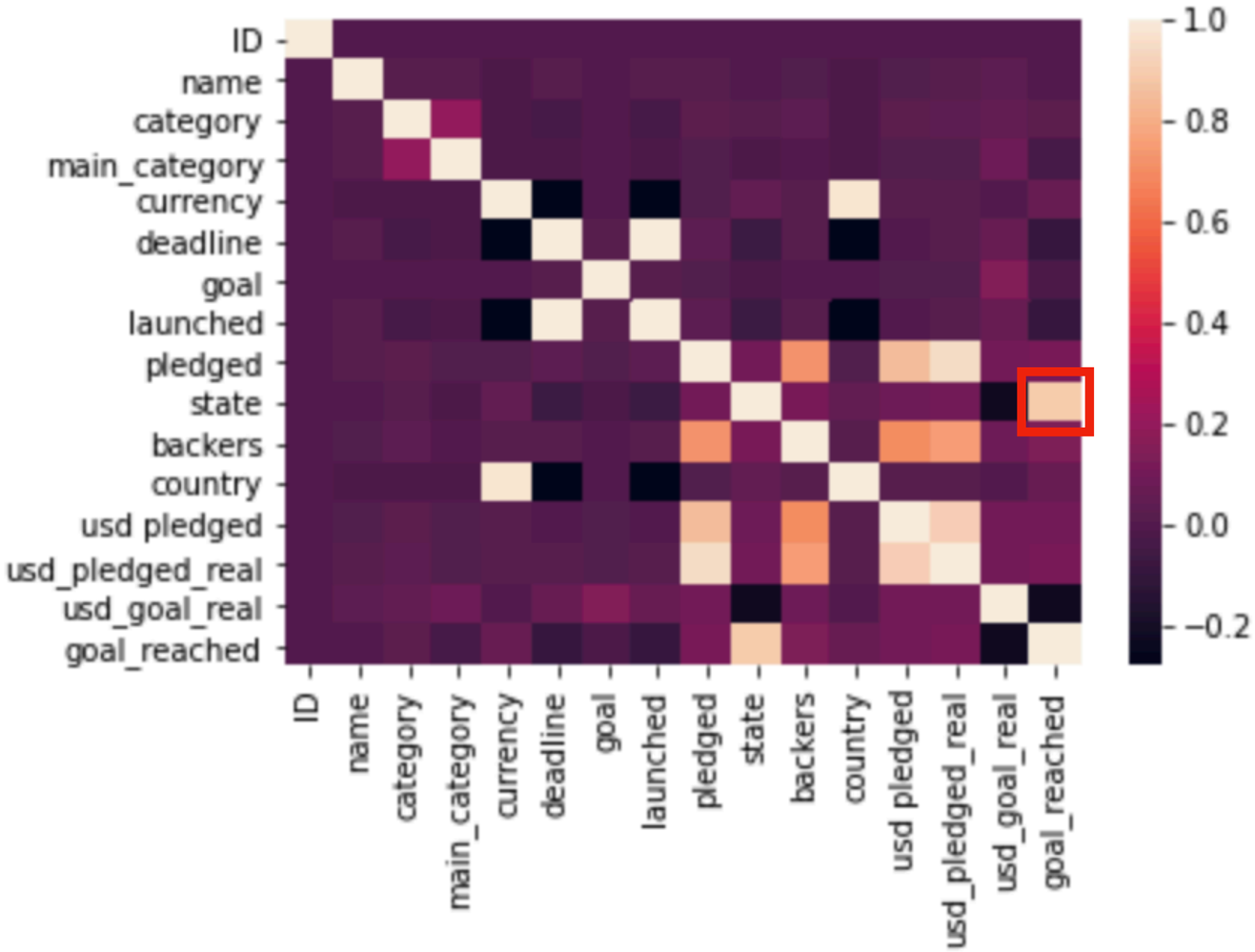
Análisis de Matriz de Correlación

Variable State Alta correlación con la variable dependiente

Durante la correlación, podemos observar algo interesante. La variable de **state**, tiene una alta correlación con nuestra nueva variable de salida **goal_reached** por lo que tenemos que analizar un poco qué es lo que está sucediendo con estás variables.

Ya que la variable **state** tiene como datos si un proyecto fue **exitoso, fallido u otro,** si no la eliminamos, esta nos puede dar falsos negativos en nuestro modelo, ya que al tener esta variable disponible, es muy probable que nos de porcentajes de precisión altos cuando no es el caso en realidad.

Al utilizar la variable de **state**, nuestros primeros modelos dieron falsos positivos, precisión, recall y f1 por arriba del 98% , aspectos que nos hice re analizar una correlación con la variable dependiente.



Eliminación de Variables Futuras

Se llevó a cabo la eliminación de las siguientes variables como se menciona en los requisitos.

- goal
- Backers
- pledged
- usd pledged
- pledged
- **usd_pledged_real** - Se cambió por nuestra nueva variable de salida

Selección de Variables dependientes

Después de llevar a cabo el análisis anterior, se seleccionaron las variables

```
name              int64
category          int64
main_category     int64
currency          int64
deadline          int64
launched          int64
country           int64
usd_goal_real   float64
```

Nueva Matriz de Correlación

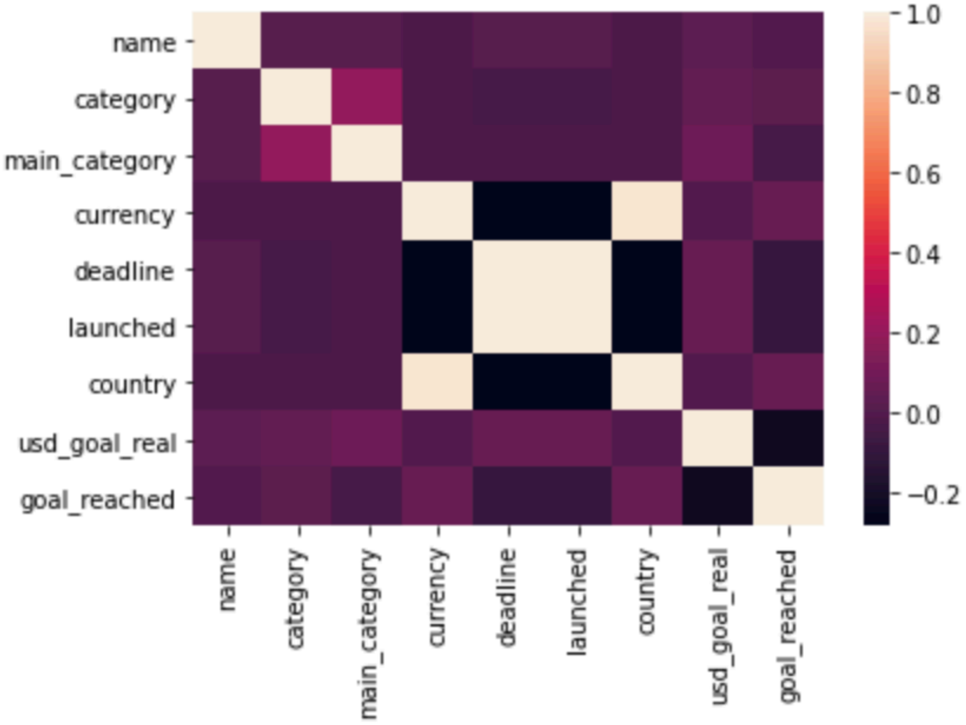


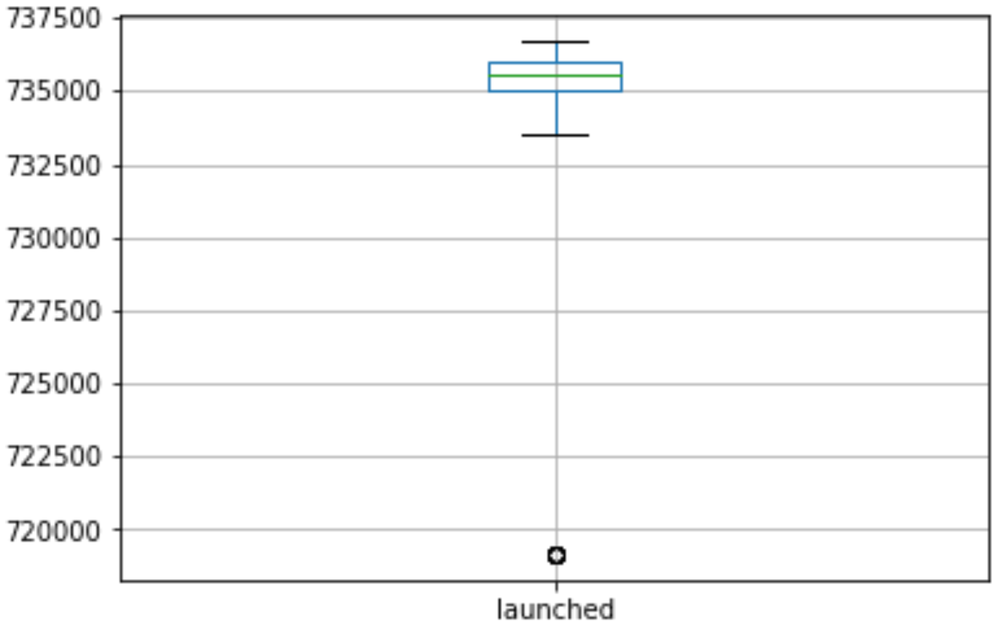
Tabla Final

	name	category	main_category	currency	deadline	launched	country	usd_goal_real	goal_reached
0	326290	108	12	5	735880	735821	9	7.977321	0
1	132984	93	6	13	736634	736574	22	11.605945	0
2	357882	93	6	13	734925	734880	22	12.119890	0
3	338194	90	10	13	734609	734579	22	9.390248	0
4	76586	55	6	13	735839	735783	22	11.065033	0

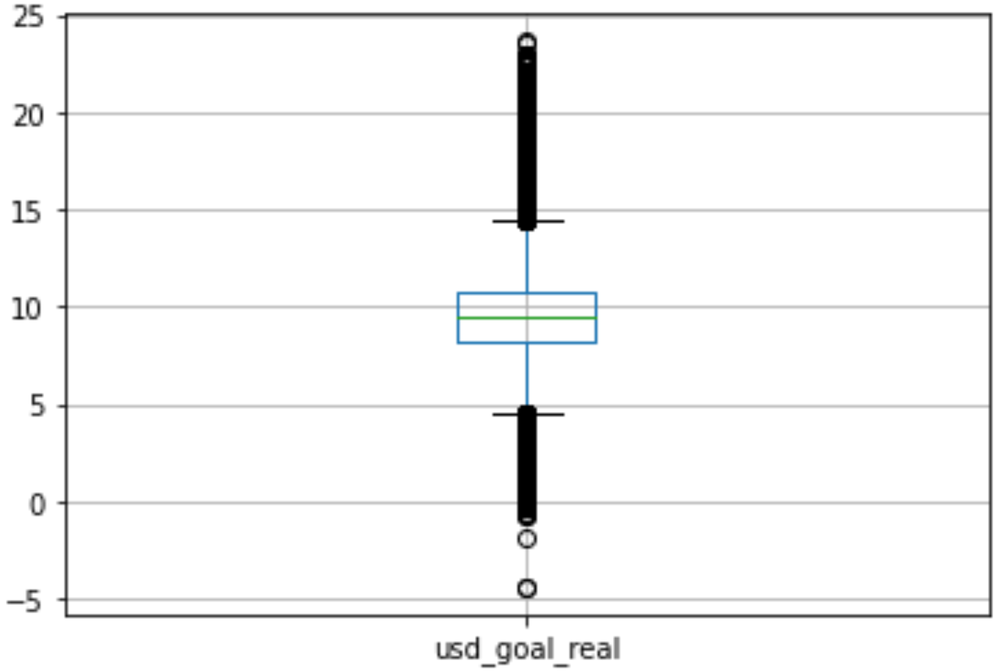
Outliers

Detectamos dos variables que tenían outliers bastante significativos.

Launched



USD Goal

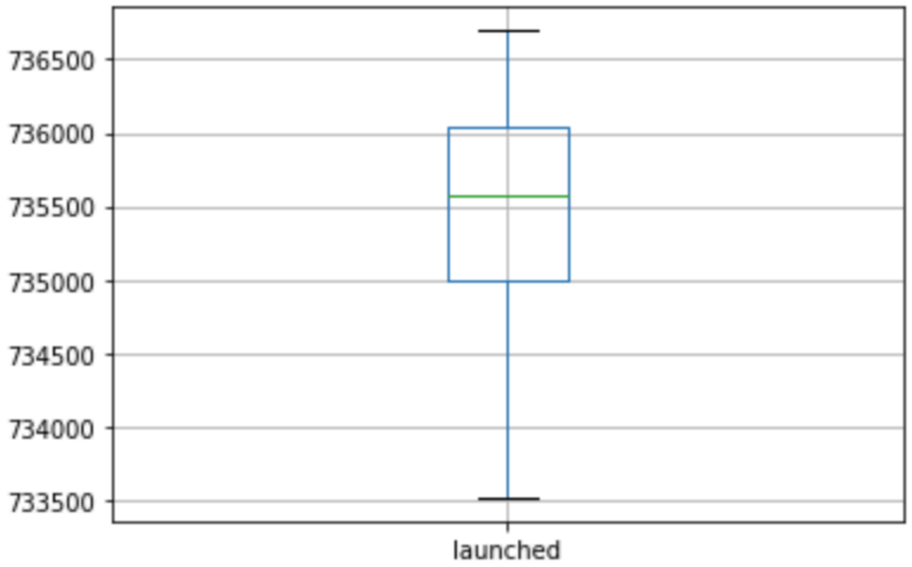


Eliminación de Outliers

Detectamos dos variables que tenían outliers bastante significativos.

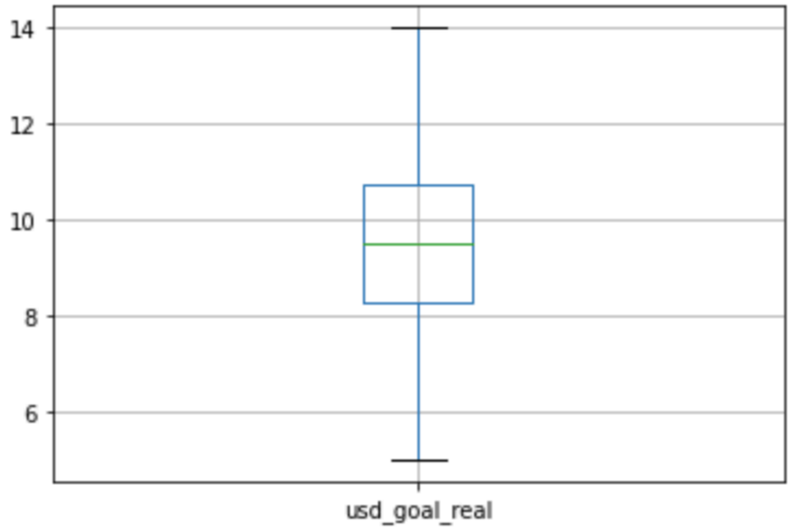
```
# Limpiamos valores que sean mayores a 73000
data = data[data["launched"] > 730000]
data.boxplot(column=["launched"])
data.shape
```

```
(378654, 9)
```




```
data = data[(data['usd_goal_real'] >= 5) & (data['usd_goal_real'] <= 14)]
data.boxplot(column=["usd_goal_real"])
data.shape
```

```
(364987, 9)
```

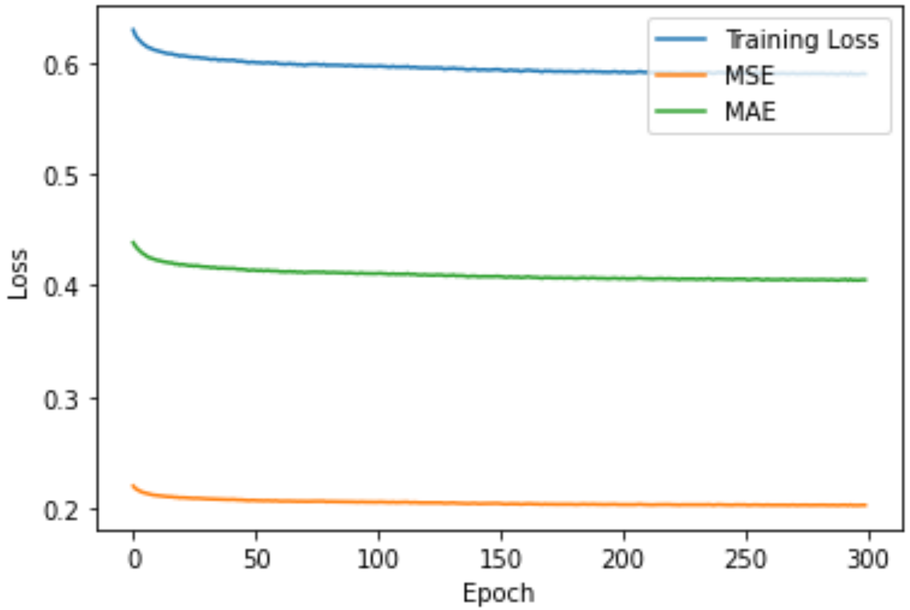


Resultados Modelo 1

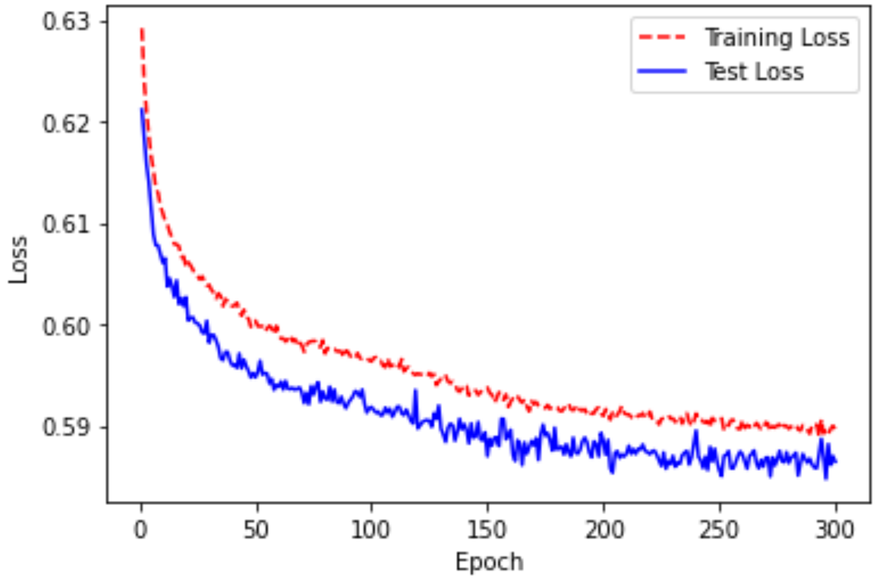
MLP

- Partición de entrenamiento 50%
- Shuffle
- Random
- Binary Cross Entropy - Ya que nuestro salidas son 0 y 1
- Optimizador Adam
- Métricas
  - Accuracy
  - MSE (Error cuadrado medio)
  - MAE (Error absoluto medio)
- 2 capas ocultas
  - Unidades: 64 (neuronas de salida)
  - Activation: Relu
  - Dropout: 0.15
- Capa de salida
  - 2 unidades
  - Softmax
- Épocas 300
- Batch Size: 64 -> Agilizar tiempos

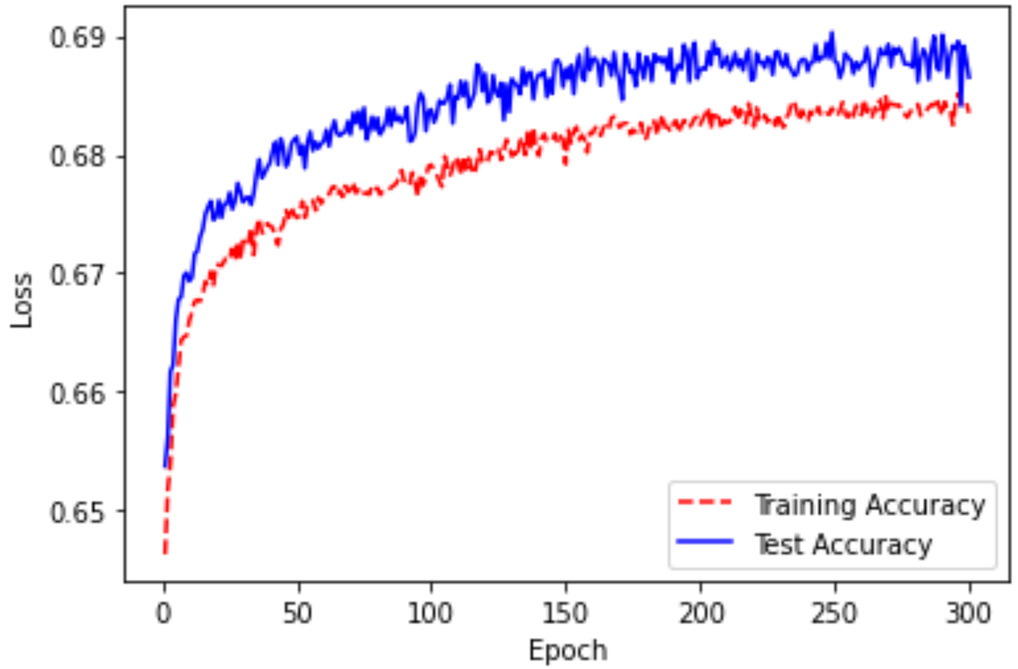
Historial Errores



Historial Función Costo



Precisión Training vs Test



RESULTADOS FINALES

	precision	recall	f1-score	support
0	0.70	0.89	0.78	115961
1	0.64	0.33	0.44	66533
micro avg	0.69	0.69	0.69	182494
macro avg	0.67	0.61	0.61	182494
weighted avg	0.68	0.69	0.66	182494
samples avg	0.69	0.69	0.69	182494

Comentarios para siguientes modelos

Se llevo a cabo una partición de 40% de entrenamiento

Se escalaron las variables de los datos originales ya que de no hacerlo se estaban obteniendo malos resultados.

En algunos modelos pudimos llevar a cabo un randomized search pero en otros casos no, como lo fue en el SVM ya que tardó muchas horas y no pudo completarse

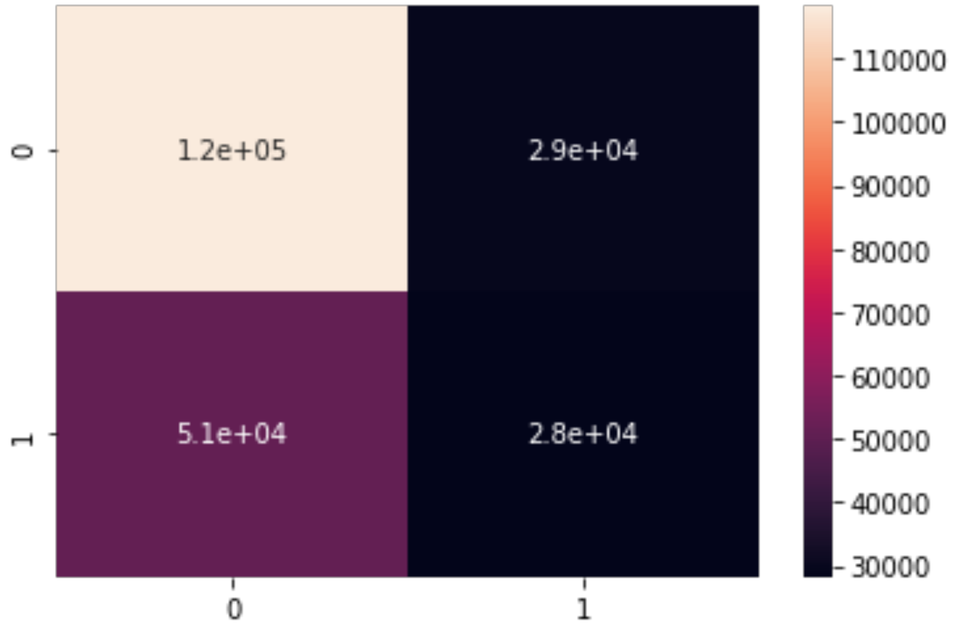
Modelo 2
KNN

- Number of Neighbors = 7
- Training 40%

Resultados

	precision	recall	f1-score	support
0	0.70	0.80	0.75	147648
1	0.49	0.36	0.41	79545
accuracy			0.64	227193
macro avg	0.59	0.58	0.58	227193
weighted avg	0.62	0.64	0.63	227193

Matriz de Confusión



Modelo 3

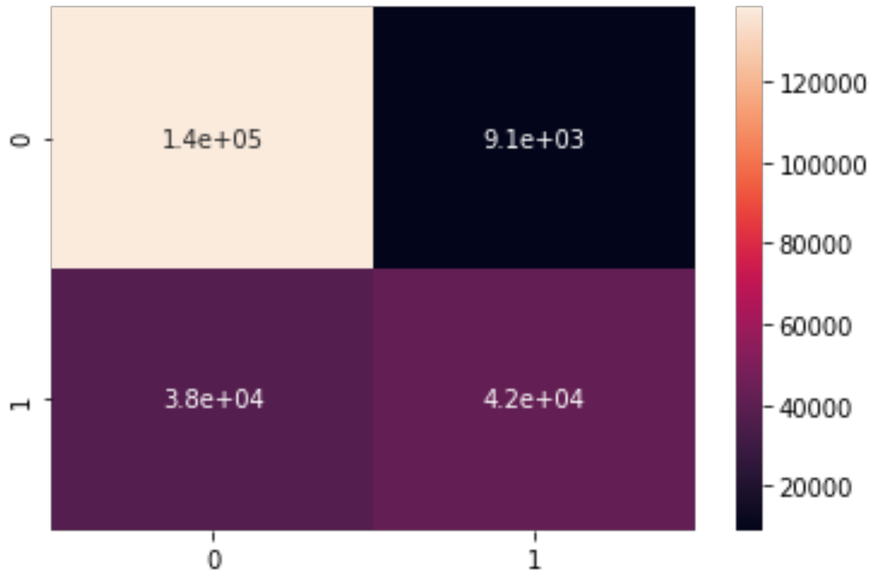
SVM

- Kernel Linear
- Bagging Classifier - Agiliza modelo y reduce bias
- Num of Estimators - 10

Resultados

	precision	recall	f1-score	support
0	0.79	0.94	0.86	147386
1	0.82	0.53	0.64	79807
accuracy			0.79	227193
macro avg	0.80	0.73	0.75	227193
weighted avg	0.80	0.79	0.78	227193

Matriz de Confusión



Modelo 4

Random Forest with Randomized Search

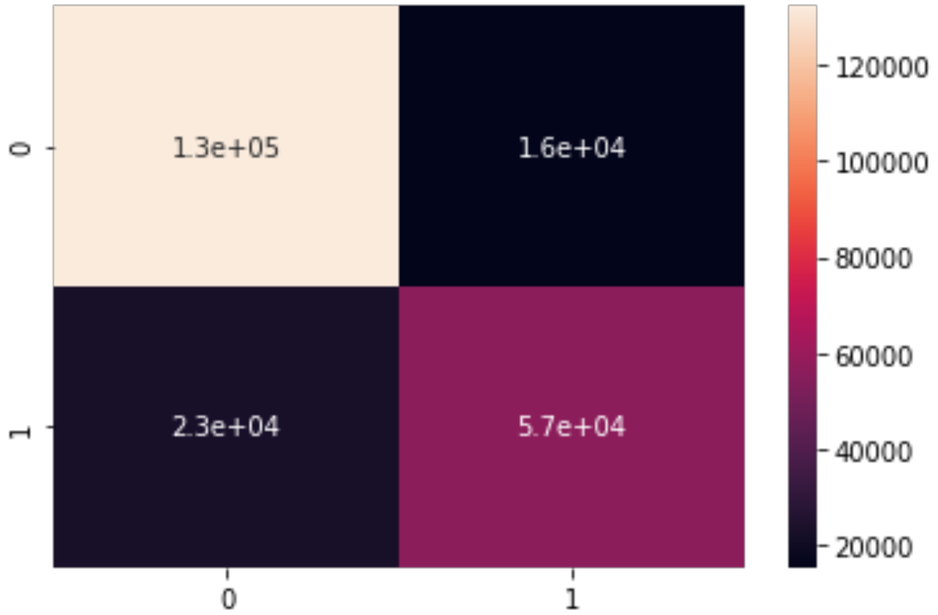
```
Bootstrap =>            False
ccp_alpha =>     0
Criterion =>     gini
Estimator =>     125
OOB Scroe =>     False
```

- Max depth 6
- Min Samples leaf - 6
- Min Samples Split - 6

Resultados

	precision	recall	f1-score	support
0	0.85	0.89	0.87	147629
1	0.78	0.71	0.75	79564
accuracy			0.83	227193
macro avg	0.82	0.80	0.81	227193
weighted avg	0.83	0.83	0.83	227193

Matriz de Confusión



Modelo 5

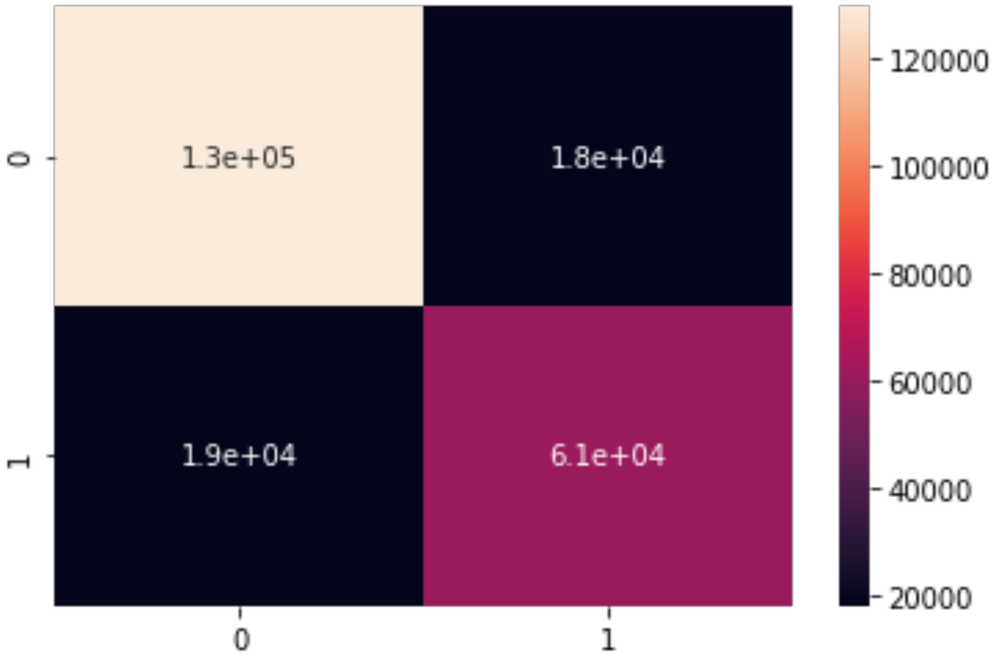
Boosted Random Forest

- Bootstrapping - True
- OOB Score - True
- Entropy
- Max depth 6
- CCP Alpha - 0
- Min Samples leaf - 6
- Min Samples Split - 6

Resultados

	precision	recall	f1-score	support
0	0.88	0.88	0.88	147629
1	0.77	0.77	0.77	79564
accuracy			0.84	227193
macro avg	0.82	0.82	0.82	227193
weighted avg	0.84	0.84	0.84	227193

Matriz de Confusión



Conclusiones

1. **Mejor Modelo: Boosted Random Forest**
2. **Boosted y Random Forest fueron los que menor tiempo tardaron y mejores resultados obtuvieron.**
3. **Sería bueno analizar SVM con Randomized Search para ver si se vuelve más preciso, por cuestión de tiempos no fue posible (Se dejó corriendo 4 horas y no terminó).**
4. **Existe más área de oportunidad para analizar con diferentes hiperparámetros estos modelos y ver si se pueden obtener mejores resultados.**
5. **La red neuronal puede volverse más precisa si se cambian el número de épocas así como el batch size, se pueden llevar futuros análisis de lo mismo.**
6. **Fue un análisis muy divertido, retador, con datos modernos y reales.**