UNIVERSIDAD GALILEO

FACULTAD DE INGENIERÍA DE SISTEMAS, INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN



**Reporte de Competencia de Kaggle**

**“House Prices”**

RODRIGO ALBERTO CORDERO ALVAREZ 1700

MARCELO ALFREDO DEL ÁGUILA MORAGA 17001380

KEVIN JOSÉ HERNÁNDEZ MARROQUÍN 17001095

INVESTIGACIÓN DE OPERACIONES I

SÉPTIMO SEMESTRE

GUATEMALA, 22 DE ARBIL DE 2020

**Índice**

**Introducción**

Con la tecnología de hoy en día, ha sido posible realizar modelos que tengan la capacidad de predecir valores o situaciones en dependencia del problema que se quiere atacar.

En todo el mundo, siempre existirán problemas de predicción. Esto se hace notar considerablemente en las empresas ya que siempre buscan tomar la mejor decisión para evitar caer en situaciones no deseables (como en bancarrota). Escenarios como estos, demuestran la importancia de la capacidad de poder predecir en base a ciertos parámetros importantes.

Un problema muy común es el de poder predecir un número dado ciertos parámetros. Este se conoce como un problema de regresión.

Un problema de regresión que se hace destacar es el problema de predecir el costo de casas, que están en venta, en base a ciertos parámetros.

Este problema se le conoce como “House Prices” y ha sido publicado en la página de Kaggle.com como un tipo de competencia para determinar quién ha logrado hacer el mejor modelo de predicción.

Nuestra tarea como grupo, es de construir un modelo muy bueno dedicado a la predicción de los costos de casas.

**Objetivos**

**Objetivo General:**

Construir un modelo robusto que sea capaz de predecir los costos de casas (en base a ciertos parámetros) y que nos permita colocarnos en una muy buena posición en la competencia de Kaggle.

**Objetivos Específicos:**

* Aplicar los conocimientos adquiridos en el curso de Investigación de Operaciones I y de Seminario Profesional I.
* Realizar análisis a los Datasets presentados para obtener una idea de la naturaleza de los datos.
* Aplicar distintos modelos para alcanzar el mejor resultado.
* Aprender nuevas técnicas para el manejo de datos y aplicarlos a nuestro problema.

**Marco Teórico**

* **Correlación:** La Correlación en la estadística determina la relación o dependencia entre dos variables, en otras palabras, determina si los cambios de una de las variables influyen en la otra variable, de ser este el caso se dice que las variables están correlacionadas.
* **Feature Engineering:** Es un proceso que se encarga de usar “domain knowledge” de la data para crear cualidades que son necesarios para que el algoritmo del machine learning funcione, si lesas cualidades son hechas de manera correcta, incrementa la certeza de las predicciones, al crear nuevas cualidades desde raw data que facilita el proceso de aprendizaje del algoritmo.
* **Integer Encoding:** Es un proceso diseñado para tomar columnas cualitativas que no tienen sus ítems no sigan un patrón de peor a mejor, por ejemplo, sino que son cualidades separadas y convertirlas en columnas con números enteros.

La manera en la que funciona es, se observa cuentas diferentes cualidades conforman a la columna, se crea una nueva columna por cada cualidad, y en esa columna se pone un “1” en todas las filas donde aparezca la cualidad que se esta analizando dentro de la columna original y se pone un “0” en las demás filas, y así hasta tener una fila por cualidad.

* **Métricas:** en este proyecto destacaron 2 métricas que dictaron lo bueno de nuestros modelos:
  + RMS (Root Mean Squared): Es un algoritmo que se encarga de dar un promedio de un valor donde no se necesita o no es importante el signo de los valores, por ejemplo si se quiere obtener el promedio de los errores de un modelo , obtenernos de una manera convencional con la fórmula de la media no nos daría un valor útil debido a que los errores pueden ser negativos o positivos , ahí el RMS nos permite obtener el promedio del error del modelo, lo cual nos permite tener una métrica para calificar nuestro modelo.
  + Mean Absolute Error: Al igual que el valor RMS el MAE nos permite obtener el promedio de las diferencias entre los valores obtenidos por el modelo y los valores reales, la diferencia entre ambas métricas es su forma de calcularla.
* **Normalización:** La normalización es un proceso en el cual se toma una columna con valores enteros o reales y se restringen a valores entre 0 y 1, lo cual nos permite reducir la carga a la computadora a la hora de procesar los valores debido a que es más fácil para la computadora trabajar con ese tipo de valores.
* **One Hot Encoding:** Al igual que Integral Encoding este proceso se encarga de tomar columnas de cualitativas y convertirlas en números para poder utilizarlas en un modelo matemático, la diferencia es que este tipo de proceso se aplica a columnas cullas cualidades tengan una relación entre ellas, por ejemplo si se tiene cualidades que van de muy malo a muy bueno se le puede asignar un numero a cada cualidad por ejemplo muy malo es 0 y muy bueno es 5 lo cual nos permite mantener el mismo nivel de cualidad pero con valores numéricos.

* **R:** Es un lenguaje y un ambiente de programación dedicado a estadística computacional y gráficos, R comparte muchas similitudes con otro leguaje de programación llamado S, R es in proyecto de GNU y S es un lenguaje desarrollado por AT&T, R proporciona una amplia gama de funciones estadísticas como herramientas graficas y un ambiente Open Source.
* **Redes Neuronales:** Es una clase de inteligencia artificial que intenta imitar la forma en la que el cerebro humano funciona, en vez de trabajar con modelos digitales , una red neuronal funciona creando conexiones entre “elementos de procesamiento”, que es el equivalente computacional de una neurona, se le asigna un peso a cada conexione entre las neuronas mientas la maquina aprende lo cual le permite mejorar sus predicciones si se entrena de la manera correcta.
* **Regresión:** Es una clase de algoritmos en estadística que se utilizan para predecir valores o resultados en dependencia de una cantidad n de variables, las regresiones pueden ser: lineales, polinómicas, logísticas, logarítmicas, entre otras.
* **Regresión Lineal:** una regresión lineal es un algoritmo de predicción que se encarda de generar una recta de predicción que se acople de la mejor manera a los valores que se tengan, una de sus características mas importantes es que la distancia entre los valores reales a la recta de predicción sumados da 0, eso nos permite obtener una sola recta de interés.
* **Tensorflow:** Es una librería open source y compatible con Python que facilita el desarrollo de Machine learning.

**Problema**

**Situación por Enfrentarse:**

El problema que nos enfrentamos, según la página oficial de Kaggle, es la predicción del valor de una casa en base a ciertos parámetros. Quieren demostrar que el precio de una casa depende más que solo el material con el que está hecho, el número de cuartos, etc.

Nosotros podemos deducir que este problema es de tipo de regresión.

En la página oficial de Kaggle, menciona una breve introducción al problema y nos proporciona ciertos archivos que nos ayudarán a resolver este problema.

Entre los archivos que nos proporciona podemos encontrar:

* *train.csv*: Archivo que contiene el dataset para el entrenamiento del modelo.
* *test.csv*: Archivo que contiene el dataset para que el modelo prediga los precios de las casas.
* *data\_description.txt*: Archivo que contiene una descripción detallada de todos los features que se encuentran en los datasets.
* *sample\_submission.csv*: Archivo que muestra un template de cómo se debe subir las predicciones a la competencia.

La descripción que nos muestra Kaggle sobre los dataset es que hay un total de 79 features y en base a ellas se quiere predecir ‘SalePrice’, que es el costo de la casa; este es el objetivo del modelo a construir.

**Problemas Potenciales**

* Poco conocimiento y poca práctica en el área de Ciencias de Datos.
* Muchas variables explicativas; hay tantas variables categóricas como cuantitativas.
* Disponemos de 1460 datos para entrenamiento, consideramos que es poco.
* No conocemos demasiadas herramientas para armar un modelo muy robusto.
* Es una competencia abierta para muchos participantes para todos los niveles, entonces será difícil conseguir una posición muy alta.

**Approaches**

Decidimos atacar el problema usando distinos approaches.

Nosotros decidimos que utilizaremos los siguientes:

* Regresiones Lineales
* Redes Neuronales
* Regresiones Lineales con XGBoost

**Regresiones Lineales**

**Regresiones Lineales con XGBoost**

**Redes Neuronales**

Las redes neuronales pueden ser modelos muy eficaces tanto para los problemas de regresión como los problemas de clasificación.

Se decidió utilizar redes neuronales debido a que un integrante del grupo tiene experiencia moderada con dichos modelos.

El reto de las redes neuronales es encontrar la mejor arquitectura que pueda realizar muy buenas predicciones.

El approach que se tomó es armar una arquitectura cuyas capas ocultas están compuestas por una cantidad de neuronas en potencias de 2 y decrecientes e ir agregando o quitando en dependencia de los resultados que proporcione la página de Kaggle.

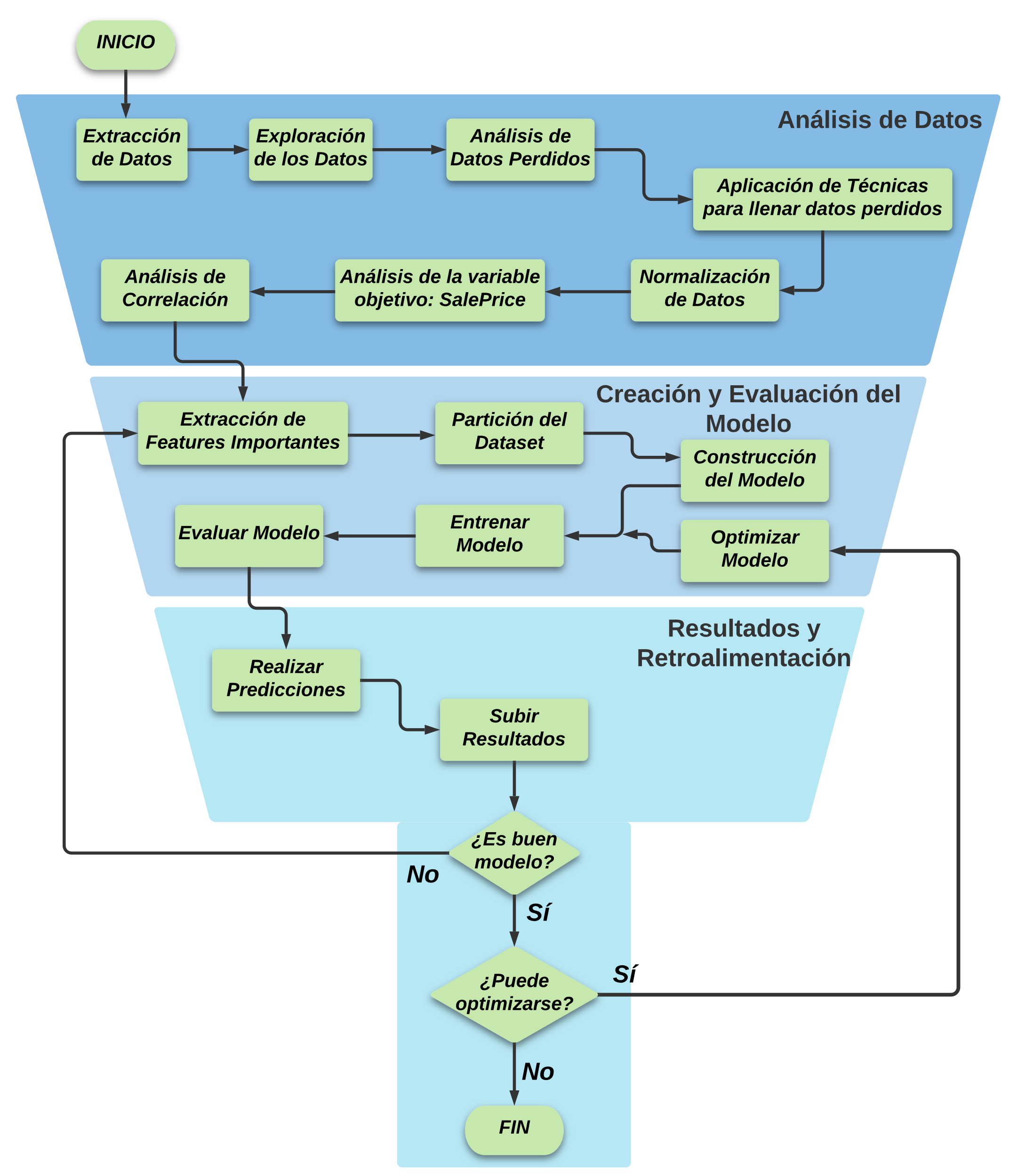
Para las redes neuronales, hay una cantidad muy grande de parámetros que se pueden modificar, sin embargo, para reducir dicha cantidad, solo se enfocará en modificar:

* El número de capas ocultas de la arquitectura.
* El número de neuronas por capa oculta.
* Inicialización de pesos de las neuronas.
* El optimizador de la red.
* La métrica de la red.
* Batch Size y Validation Split.
* Número de Epochs.

Para evaluar qué tan buena es la red, se utiliza la métrica de Loss.

**Metodología**

La metodología utilizada para la realización de este proyecto es el siguiente:



**Detalles de la Metodología**

**Fase de Análisis de Datos**

1. Extracción de Datos:

Consiste en extraer los datos contenidos en los archivos *train.csv* y *test.csv* y construir los datasets de train y test, respectivamente.

1. Exploración de los Datos:

Consiste en conocer ambos datasets (train y test) y observar qué features tiene cada uno, el tipo de dato de cada feature (si es categórico o cuantitativo) y cuántos datos tiene cada dataset. Se decidió combinar ambos datasets en uno solo para analizar los datos.

En este paso se utilizó mucho el archivo *data\_description.txt* para ir analizando qué significa cada feature y los posibles valores que podía tener.

1. Análisis de Datos Perdidos:

Consiste en conocer qué features tienen “Missing Values”, valores Nulos o valores de tipo NaN. Se consultó muy frecuentemente el archivo *data\_description.txt*, ya que especificaba que algunos features que poseen Missing Values o NaN sí representaban un valor como tal.

Después de haber hecho el análisis descrito anteriormente, se procede a aplicar técnicas[1] para el reemplazo y rellenado de Missing Values o NaNs.

1. Encoding:

Consiste en aplicar las técnicas de Integer Encoding o One Hot Encoding a cada feature de tipo categórico, según sea necesario.

1. Normalización de Datos:

Consiste en normalizar los datos de cada feature para disminuir el tiempo de entrenamiento del modelo y mejorar su capacidad para predecir.

1. Análisis de la variable objetivo ‘*SalePrice*’:

Consiste en realizar un análisis estadístico a la variable SalePrice:

* Analizar su distribución
* El sesgo de la gráfica de distribución
* Media
* Varianza

1. Análisis de Correlación:

Consiste en analizar la correlación de todos los features con la variable objetivo: SalePrice.

Además, se realiza un mapa de calor de correlación para facilitar la visualización de las correlaciones entre los distintos features.

1. Separación del Dataset:

Consiste en separar el dataset combinado (en el dataset de train y test) después de haber terminado el análisis de datos.

**Fase de Creación y Evaluación del Modelo**

1. Extracción de Features Importantes:

En base al análisis de correlación, se extraen los features que cumplen con el criterio de “Threshold”, es decir, si la correlación de un determinado feature con SalePrice es igual o mayor a un Threshold establecido, entonces es considerado como un feature importante.

Por último, se crea un nuevo dataset ‘*train’* compuesto únicamente por el conjunto de features extraídos anteriormente.

1. Partición del Dataset:

Consiste en hacer un shuffle al nuevo dataset y particionarlo en 2 nuevos datasets: *newTrain* y *evaluation*. El propósito del dataset *newTrain* es ser utilizado para el entrenamiento del modelo, y el propósito del dataset *evaluation* es para evaluar qué tan bueno es nuestro modelo. Se utiliza un criterio de 80% de los datos para newTrain y 20% para *evaluation* y

1. Construcción del Modelo:

Consiste en crear el modelo ya sea usando una regresión lineal, regresión XGBoost o una red neuronal.

1. Entrenar Modelo:

Consiste en utilizar el dataset *newTrain* para entrenar el modelo construido anteriormente.

1. Evaluar Modelo:

Consiste en utilizar el dataset *evaluation* para evaluar el desempeño del modelo.

Para los modelos de regresión lineal o regresión XGBoost se utilizó la métrica RMSE para evaluar el desempeño.

Para los modelos de tipo red neuronal, se utilizó la métrica Mae y Loss, y en algunos casos RMS y Loss.

**Fase de Resultados y Retroalimentación**

1. Realizar Predicciones:

Consiste en utilizar el modelo para predecir la variable objetivo (SalePrice) para los distintos datos contenidos en el dataset *test*.

1. Subir Resultados:

Consiste en crear el archivo con el formato especificado en Kaggle (usando las predicciones obtenidas anteriormente) para luego subir nuestras predicciones y así obtener un score y avanzar en la tabla de Leaderboard.

1. (Paso Extra) Optimizar Modelo:

Luego de obtener el respectivo punteo en Kaggle, se decide si vale la pena optimizar el modelo cambiando ciertos parámetros según sea el modelo.

**Hallazgos**

**Fase de Análisis de Datos**

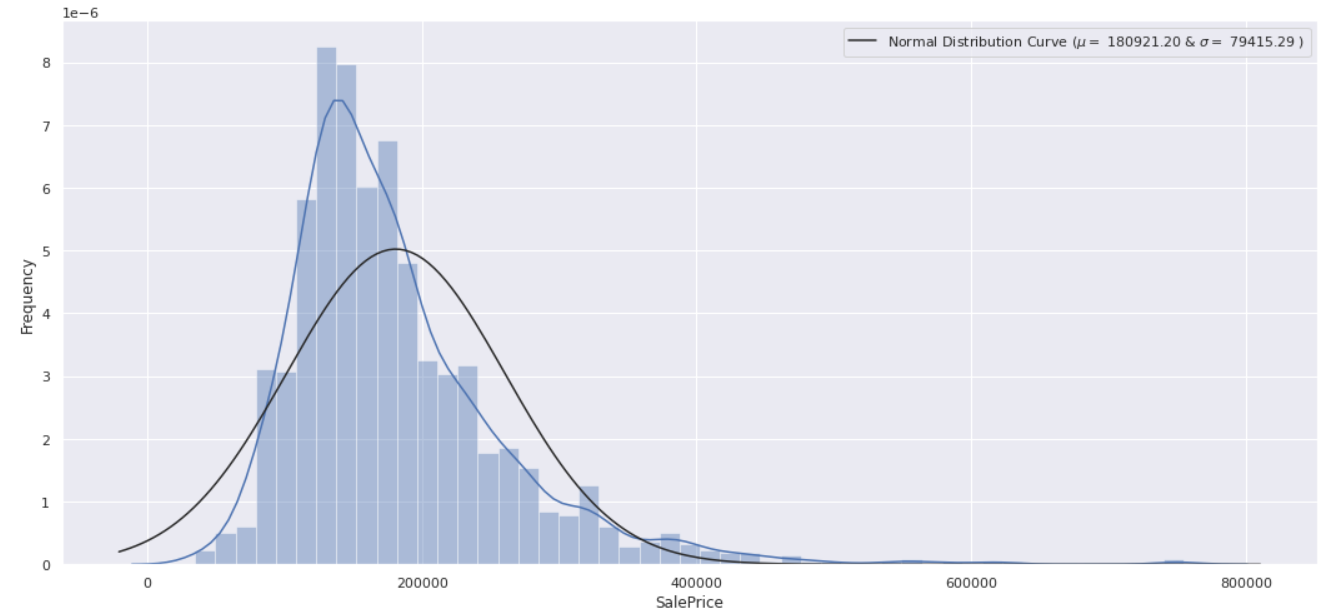
* Análisis de Datos Perdidos:

Se encontraron varios features que poseen Missing Values o NaNs. Sin embargo, el archivo no describía que dichos features tenían la posibilidad tener un valor NaN o un “Missing Value”.

Para las técnicas de reemplazo y rellenado de Missing Values o Nans, se decidió que:

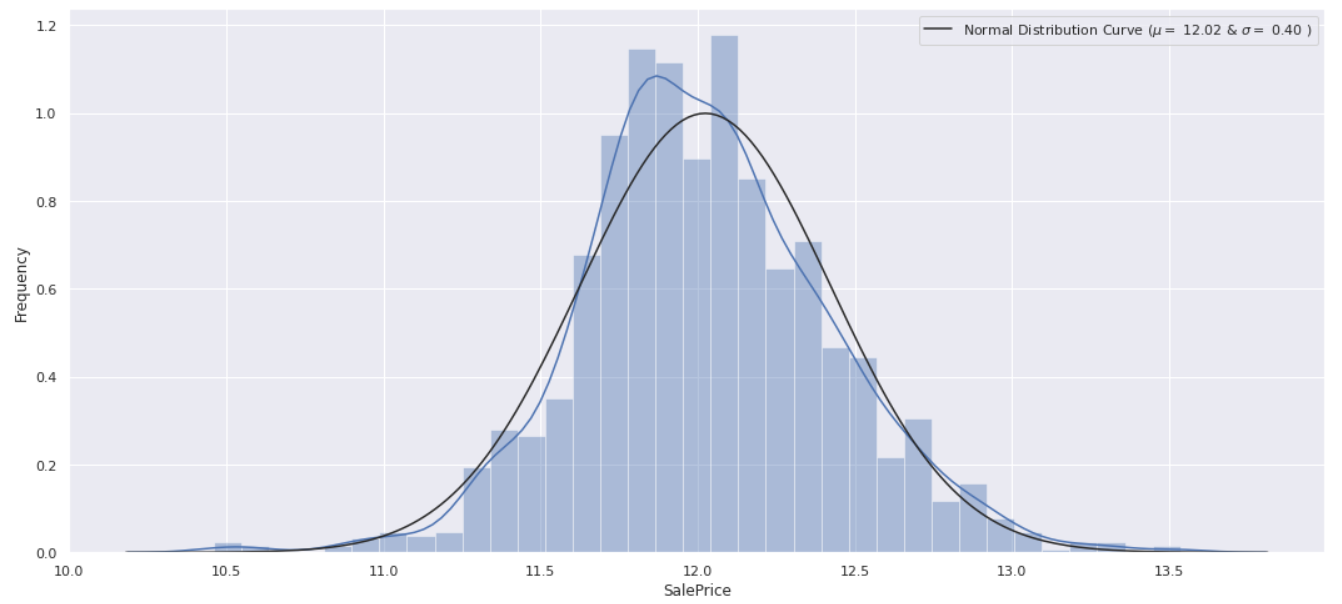
* + Para el feature *‘LotFrontage’,* los valores NaN son reemplazados por la mediana del feature*.*
  + Para los Features *‘GarageyrBlt’*, *‘GarageArea’*, *‘GarageCars’*, *‘BsmtFinSF1’*, *‘BsmtFinSF2’*, *‘BsmtUnfSF’*, *‘TotalBsmSF’*, *‘BsmtFullBath’*, *‘BsmtHalfBath’*, *‘BsmtFinSF1’* y *‘MasVnrArea’*, los valores NaN son reemplazados por un 0.
  + Para el feature *‘MSZoning’*, los valores NaN son reemplazados por la moda del feature.
* Análisis de la variable objetivo ‘*SalePrice*’:

Se descubre que la gráfica de distribución de frecuencias de la variable SalePrice se encuentra sesgada hacia la izquierda, además se descubre que dicha gráfica tiene una curtosis de tipo Leptocúrtica.



***Gráfica de Distribución de Frecuencias de SalePrice***

A la variable objetivo se le aplica una transformación de tipo Log(p+1) con el fin de normalizar la data.



***Gráfica de Distribución de Frecuencias de SalePrice luego al aplicarle la transformación Log(p+1)***

**Resultados**

**Resultados de todos los modelos que realizamos**

**Resultados del mejor modelo que realizamos**

**Interpretación de los Resultados**

**Conclusiones**

* El paso fundamental es hacer un análisis del dataset previo a construir los modelos.
* Es muy importante realizar un análisis a todas las variables involucradas para obtener una idea de la naturaleza de los datos.
* Es necesario aplicar técnicas de ‘Feature Engineering’ y de limpieza de datos para mejorar el desempeño de los modelos a construir.
* Es difícil encontrar cómo ‘mejorar’ un modelo y asegurar que esa ‘mejora’ sea algo positivo y no sea perjudicial.
* El desempeño de las redes neuronales decrece conforme se aumenta la complejidad del mismo.

**Observaciones**

**Recomendaciones**

**Anexos**

* **Repositorio de GitHub:**

<https://github.com/corderobot/Kaggle_HousePrices>

**Referencias Bibliográficas**

# 

**There are no sources in the current document.**