# INFORME PRÁCTICO TRABAJO FINAL

WILMER ALEXANDER ESCOBAR BOTINA
YAMID ESTEBAN ESTRADA BASTIDAS
PABLO ANDRES VACA

ELECTIVA APRENDIZAJE AUTOMÁTICO
MISC - MGTIC
UNIVERSIDAD DE NARIÑO
2021

# INFORME PRÁCTICO TRABAJO FINAL

WILMER ALEXANDER ESCOBAR BOTINA
YAMID ESTEBAN ESTRADA BASTIDAS
PABLO ANDRES VACA

PRESENTADO A: MSc. JIMMY MATEO GUERRERO RESTREPO

ELECTIVA APRENDIZAJE AUTOMÁTICO
MISC - MGTIC
UNIVERSIDAD DE NARIÑO
2021

TABLA DE CONTENIDO

# INTRODUCCIÓN

En la actualidad el aprendizaje automático ha ido ganando protagonismo en diversos campos y áreas de investigación, tales como la medicina, comercio electrónico, finanzas, economía, venta de bienes raíces, entre otros. Los resultados obtenidos después de someter las grandes cantidades de datos a técnicas de *Machine Learning* han mejorado considerablemente debido a la aplicación de herramientas de descubrimiento de patrones, modelos y tendencias.

Puntualmente en el sector inmobiliario, el aprendizaje automático se ha convertido en una pieza clave para lograr estimaciones bastante aproximadas a la realidad sobre precios de compra y venta de inmuebles teniendo en cuenta variables como ubicación, tamaño, características, tipo de inmueble, entre otras que pueden afectar directamente los costos y la rentabilidad de un bien raíz.

Gracias a la integración de algoritmos, técnicas, librerías, plugins con diversos lenguajes de programación como Python o R Studio es posible llevar a cabo el procesamiento de grandes cantidades de datos en distintos formatos para ser entrenados y poder generar modelos de clasificación y predicción a partir de los mismos. En este informe se presenta un caso de estudio en donde se busca desarrollar un modelo predictivo del valor del metro cuadrado de inmuebles en la ciudad de Buenos Aires.

El informe consta de un apartado donde se presenta el caso de estudio, después está la etapa de preparación de los datos, continuando con el modelado, la implementación, evaluación y finalmente se presentan las conclusiones.

# CASO DE ESTUDIO

La Inmobiliaria *www.barealestate.com.ar* dispone de un sitio Web para la compra y venta de inmuebles en la Ciudad de Buenos Aires. Para disminuir los costos de publicación, se necesita disponer de una pre valoración de los inmuebles publicados a la venta por los propietarios. La empresa necesitaría de alguna forma estimar el rango del precio del m2 del inmueble para decidir el nivel de especialista avaluador que destinará para realizar una inspección física y así obtener una cotización real del precio de venta.
Para la empresa, el costo de la publicación depende mucho del nivel del especialista avaluador que tiene que realizar la cotización, por lo tanto, desea bajarlo manteniendo acotado el riesgo de una mala cotización. La empresa decidió realizar una competencia abierta para obtener un modelo predictivo a partir de sus datos históricos para obtener una estimación de precio del *m2*.

Para decidir el nivel de especialista para tasar el inmueble es suficiente conocer el precio del m2 sólo en los siguientes tramos:

Clase 1: m2 <= $2.000 2.
Clase 2: $2.000 < m2 <= $2.500
Clase 3: $2.500 < m2 <= $5.500
Clase 4: $5.500 < m2 <= $18.500
Clase 5: $18.500 < m2

El objetivo de este trabajo práctico es desarrollar un modelo de predicción del rango del precio en m2 de propiedades a la venta en la Ciudad de Buenos Aires.

# PREPARACIÓN DE LOS DATOS

Inicialmente se trabaja con 2 set de datos en formato CSV, uno para el entrenamiento del modelo y otro para la entrega de la clasificación denominados: **"tpFinal-entrenamiento.csv"** con un total de 12000 registros y **"tpFinal-clasificacion.csv"** que tiene 3000 registros respectivamente. Los atributos que contienen los archivos son:

Tabla 1. Atributos iniciales

| Atributo | Descripción | Tipo |
|---|---|---|
| id | Identificador único de fila | Numérico |
| fecha | Fecha de venta | Fecha |
| anio | Año de venta | Numérico |
| mes | Mes de venta | Texto |
| tipoprop | Tipo de propiedad(casadto,ph,local) | Texto |
| lugar | Zona geográfica(todas las localidades de Caba) | Texto |
| geoname_num | GeoName id | Numérico |
| sup_tot_m2 | Superficie total en metros cuadrados | Numérico |
| sup_cub_m2 | Superficie cubierta en metros cuadrados | Numérico |
| Piso | Piso del inmueble si corresponde | Numérico |
| cant_amb | Cantidad de ambientes del inmueble | Numérico |
| descripción | Descripción de la publicidad de venta | Texto |
| lat | Latitud del inmueble | Real |
| lon | Longitud del inmueble | Real |
| Clase | Discretización de usd m2 | Numérico |

En el archivo **"tpFinal-clasificacion.csv"** el atributo Clase no estará y será completado con el modelo de clasificación obtenido al final de este informe.

A partir del dataset de entrenamiento se realiza una revisión de valores vacíos y/o nulos y se definen distintas estrategias y transformaciones de relleno y limpieza de los distintos atributos, adición de nuevos atributos o eliminación de variables consideradas irrelevantes, que van a permitir el aprendizaje y entrenamiento del modelo para una mejor clasificación de los datos.
De acuerdo con el orden de los atributos mencionados anteriormente, se procede eliminando valores nulos de la siguiente manera:

- Para el campo **geoname_num** se busca y reemplaza los valores faltantes en la página Geonames[1] a partir de la columna lugar dentro de la ciudad de Buenos Aires. En otros casos se busca el nombre formal de los lugares o de sitios con mayor cobertura que los rodean.
- En la columna **sup_tot_m2** se ubica el valor de **sup_cub_m2** cuando éste último no es nulo.
- En la columna **sup_cub_m2** se ubica el valor de **sup_tot_m2** cuando éste último no es nulo.
- En el caso en que los 2 valores sean nulos, se calcula el valor de **sup_tot_m2** y **sup_cub_m2** promedio de estos atributos de los inmuebles del mismo lugar.
- Para la columna **Piso** cuando el valor es mayor a 60 se divide entre 100 con el fin de evitar cantidades atípicas. Con los valores nulos de ese atributo en donde se tienen información del lugar, se reemplaza con el promedio. Para los valores restantes en donde no se tiene información, se reemplaza con "1".
- Para el atributo **cant_amb** se aplicaron diversas estrategias para obtener la cantidad de ambientes a partir del atributo descripción. Se utilizaron expresiones regulares con el fin de abarcar la mayor cantidad de variaciones o ubicaciones dentro del texto: en el comienzo, antes de un espacio o simplemente seguido de la sílaba "amb" y demás variaciones como "ambi", "ambient", "ambiente". Se validan los valores de 2 dígitos y cuando la cantidad se encuentra en letras. También se revisa que la cantidad esté después de las palabras mencionadas anteriormente. En los casos donde no se encuentra la raíz "amb" y sus variaciones se le asigna "1" como valor por defecto.
- Los atributos **lon** y **lat** se reemplazan teniendo como referencia el atributo distancia_minima_subte, que se definirá en próximos párrafos de este informe. El valor para los campos nulos es el máximo en el que dicha distancia sea menor a 50000 del mismo lugar. Para los valores inconsistentes que provocaron que el valor de la distancia sea extremadamente grande, se ajustan teniendo en cuenta los mismos criterios.

Siguiendo con el proceso de preparación de los datos para el entrenamiento, se analizaron las variables que eventualmente podrían aportar valor al momento de predecir la clasificación. Se agregaron los siguientes set de datos en formato CSV

extraídos de la página[2] de datos públicos generados, guardados y publicados por el Gobierno de la Ciudad de Buenos Aires:

Tabla 2. Datasets información adicional

| **Dataset (.csv)** | **Descripción** |
|---|---|
| barrios | Límites y ubicación geográfica de los barrios de la Ciudad. |
| subte | Ubicación geográfica de las líneas y estaciones de subte. |
| espacio-verde-publico | Límites y ubicación geográfica de los espacios verdes de la Ciudad. |
| espacios-culturales | Listado de espacios culturales públicos, privados e independientes localizados en la Ciudad de Buenos Aires. |
| hospitales | Ubicación geográfica de los hospitales de la Ciudad de Buenos Aires. |
| iglesias | Información y ubicación geográfica de iglesias de la Ciudad. |
| oferta-gastronomica | Listado de restaurantes y otros establecimientos de oferta gastronómica existentes en la Ciudad Autónoma de Buenos Aires. |
| universidades | Listado con ubicación geográfica y datos de contacto de las Universidades en la Ciudad. |
| precio-venta-deptos | Precio promedio del m2 (dólares) de departamentos en venta de 2 y 3 ambientes usados y a estrenar. |

A partir de estos datasets se generan nuevas columnas para el entrenamiento:

- El atributo **distancia_minima_subte** es la distancia mínima del inmueble a una estación de subterráneo.
- El atributo **distancia_minima_hospitales** es la distancia mínima del inmueble a un hospital.
- El atributo **distancia_minima_culturales** es la distancia mínima del inmueble a un espacio cultural.
- El atributo **distancia_minima_publicos** es la distancia mínima del inmueble a un espacio público.
- El atributo **distancia_minima_iglesias** es la distancia mínima del inmueble a una iglesia.
- El atributo **distancia_minima_gastronomicos** es la distancia mínima del inmueble a un sitio con oferta gastronómica.
- El atributo **distancia_minima_universidades** es la distancia mínima del inmueble a una universidad.

Los siguientes atributos corresponden a la cercanía del inmueble con el sitio asignado anteriormente.

- Los nombres de los atributos se definieron como **proximo_subte**, **proximo_hosp**, **proximo_cult**, **proximo_publ**, **proximo_igle**, **proximo_gast**, **proximo_univ** y se clasificaron de la siguiente manera:
  - **MUY CERCA:** distancia menor a 200 metros.
  - **CERCA:** distancia comprendida entre 200 y 1000 metros.
  - **LEJOS:** distancia mayor a 1000 metros.
- Se adiciona el atributo **comuna** que hace referencia al número de la comuna a la que pertenece el lugar donde está ubicado el inmueble.
- El atributo **bueno_para_vivir** almacena un valor categórico ('SI', 'NO') basado en un estudio asociado a las condiciones socioeconómicas del barrio en donde está ubicado el inmueble.
- El atributo **precio_prom** fue obtenido del dataset que contiene información de diversos precios promedio de venta de inmuebles diferenciados por barrio, año, cantidad de ambientes y estado.
- Se adiciona el atributo **propiedad** que reemplaza el valor "dto" por la cadena "departamento" puesto que se consideró podría contribuir a la inferencia de la información.

Entre las variables que se consideraron irrelevantes en el proceso de entrenamiento y por lo tanto fueron eliminadas se tienen las siguientes: id, fecha, mes, anio, lugar, descripcion.

Adicionalmente se aplica minería de texto mediante el modelo TF-IDF para los atributos **descripcion** y **lugar**. En la siguiente imagen se observan los parámetros utilizados para generar los atributos que son adicionados al dataset de entrenamiento.

Figura 1. Parámetros del modelo TfidfVectorizer

```
modelo_tf_idf = TfidfVectorizer(max_df=0.6, min_df=2,
                    lowercase=True,
                    ngram_range = (1,1),
                    analyzer = 'word',
                    max_features=100)

matrix_tf = modelo_tf_idf.fit_transform(datos.tokenizar_lema.apply(lambda x: np.str_(x)))


datos_estructurado=pd.DataFrame(matrix_tf.toarray(),columns=modelo_tf_idf.get_feature_names(),index=datos.id)
```

Dentro de este proceso se realiza eliminación de acentos, palabras muertas, caracteres especiales, números, espacios en blanco.

## MODELADO

Después de la etapa de preparación, se continúa con el entrenamiento de los datos haciendo uso de diversas técnicas de aprendizaje supervisado. Una vez se tiene el conjunto de datos terminado, sin valores nulos, se aplica la función **get_dummies** que convierte en columnas cada valor distinto de cada celda. Se separan los datos que se usarán para el entrenamiento (70%) y prueba (30%) de los distintos modelos. A continuación se muestra el listado de variables, 209 en total.

Figura 2. Variables del dataset

```
['geoname_num', 'sup_tot_m2', 'sup_cub_m2', 'piso_x', 'cant_amb', 'lat', 'lon', 'distancia_minima_subte', 'distancia_minima_hos
pitales', 'distancia_minima_culturales', 'distancia_minima_publicos', 'distancia_minima_iglesias', 'distancia_minima_gastronomi
cos', 'distancia_minima_universidades', 'comuna', 'abierto', 'air', 'aire', 'almagro_x', 'alto', 'ambient', 'ambiente', 'amenit
ies', 'amplio', 'apto', 'argentino', 'avenida', 'bajo', 'balcn', 'balcon', 'bao', 'barrio_x', 'baulera', 'belgrano_x', 'caballi
to_x', 'calle', 'capital_x', 'casa', 'cochera', 'cochero', 'cocina', 'codigo', 'comedor', 'completo', 'contrafrente', 'corrid
o', 'crespo_x', 'cuadra', 'cubierto', 'cucicbar', 'cuota', 'departamentir', 'departamento', 'dependencia', 'depto', 'diario',
'doble', 'dormitorio', 'duplex', 'edificio', 'emprendimiento', 'entrada', 'estrenar', 'excelente', 'expensa', 'federal_x', 'flo
r_x', 'frente', 'hall', 'hermoso', 'impecable', 'independiente', 'inmobiliario', 'integrado', 'juan', 'lavadero', 'living', 'lo
cal', 'luminoso', 'metro', 'monoambiente', 'oportunidad', 'palermo_x', 'parque_x', 'parrilla', 'patio', 'piso_y', 'placard', 'p
lanta', 'precio', 'profesional', 'propiedad_y', 'publicado', 'reciclado', 'renta', 'salida', 'semipiso', 'servicio', 'solariu
m', 'subte', 'suite', 'terraza', 'terrazar', 'tipo', 'toilette', 'torre', 'ubicacion', 'ubicado', 'unidad', 'venta', 'ventar',
'villa_x', 'vista', 'visto', 'zona', 'abasto', 'agronomia', 'almagro_y', 'avellanedar', 'balvanero', 'barraca', 'barrio_y', 'be
lgrano_y', 'boca', 'boedir', 'caballito_y', 'canita', 'capital_y', 'castro', 'centenario', 'centro', 'cha', 'chacabucir', 'chac
aritar', 'chico', 'coghlar', 'colegial', 'congreso', 'constitucion', 'crespo_y', 'cristobal', 'devoto', 'federal_y', 'flor_y',
'floresta', 'hollywood', 'linier', 'lugano', 'luro', 'madero', 'matadero', 'microcentro', 'mitre', 'monserrat', 'monte', 'nicol
os', 'norte', 'nunez', 'once', 'ortuzar', 'palermir', 'palermo_y', 'parque_y', 'paternal', 'patricio', 'pompeya', 'puerto', 'pu
eyrredon', 'real', 'recoletar', 'retiro', 'rita', 'saavedra', 'santo', 'sarsfield', 'soho', 'soldati', 'telmo', 'tribunal', 'ur
quizo', 'velez', 'versall', 'viejo', 'villa_y', 'villar', 'tipoprop_casa', 'tipoprop_dto', 'tipoprop_local', 'proximo_subte_LEJ
OS', 'proximo_subte_MUY CERCA', 'proximo_hosp_LEJOS', 'proximo_hosp_MUY CERCA', 'proximo_cult_LEJOS', 'proximo_cult_MUY CERCA',
'proximo_publ_LEJOS', 'proximo_publ_MUY CERCA', 'proximo_igle_LEJOS', 'proximo_igle_MUY CERCA', 'proximo_gast_LEJOS', 'proximo_
gast_MUY CERCA', 'proximo_univ_LEJOS', 'proximo_univ_MUY CERCA', 'bueno_para_vivir_SI', 'precio_prom_BAJO', 'precio_prom_MEDI
O', 'precio_prom_MUY BAJO', 'propiedad_x_casa', 'propiedad_x_departamento', 'propiedad_x_local']
```

Se entrenan los datos por medio de diversas técnicas como árboles de decisión, redes neuronales, ensambles como Random Forest o XG Boost. En la siguiente sección se muestra información detallada de la implementación.

## IMPLEMENTACIÓN

En esta etapa se realizan las configuraciones de las técnicas utilizadas para generar los modelos, asignando valores a los diversos parámetros disponibles, procurando realizar la combinación más óptima para el set de datos. A continuación se muestran los parámetros de cada técnica.

Figura 3. Parámetros DecisionTree

```
from sklearn import tree
params={'ccp_alpha': 0, 'criterion': 'gini', 'max_depth': 50, 'min_samples_leaf': 5}
mejor_arbol = tree.DecisionTreeClassifier(**params)
```

Figura 4. Parámetros SkLearn

```
# Creamos el objeto del modelo de red neuronal multicapa.
red = sk.neural_network.MLPClassifier(solver='adam',
                                      learning_rate_init=lr,
                                      hidden_layer_sizes=tuple(nn[1:]),
                                      verbose=True,
                                      n_iter_no_change=500,
                                      batch_size = 124)
```

Figura 5. Parámetros Random Forest

```
from sklearn.ensemble import RandomForestClassifier

# TODO: Train the supervised model on the training set using .fit(X_train, y_train)

modelo_rf=RandomForestClassifier(**{'n_estimators':1000})

modelo_rf=modelo_rf.fit(X_train, y_train)
```

Figura 6. Parámetros XG Boost

```
params = {'booster': 'gbtree',
 'eta': 0.03,
 'eval_metric': 'mlogloss',
 'gamma': 0.03,
 'max_depth': 20,
 'n_estimators': 1000,
 'objective': 'binary:logistic',
 'scale_pos_weight': 2.682472778363189,
 'use_label_encoder': True}


modelo_xgb = XGBClassifier(**params)
```

Los resultados de la parametrización realizada y evaluación de las técnicas empleadas son:

Figura 7. Resultados evaluación DecisionTree

```
accuracy test: 0.42916666666666664
accuracy train: 0.7532142857142857
Balanced_accuracy test: 0.42307280244731793
Balanced_accuracy train: 0.7483328984730397
              precision    recall  f1-score   support

           1       0.41      0.50      0.45       805
           2       0.37      0.31      0.34       596
           3       0.49      0.46      0.47       720
           4       0.48      0.46      0.47       767
           5       0.39      0.38      0.39       712

    accuracy                           0.43      3600
   macro avg       0.43      0.42      0.42      3600
weighted avg       0.43      0.43      0.43      3600
```

Figura 8. Resultados evaluación SkLearn

```
accuracy test: 0.22361111111111112
accuracy train: 0.23547619047619048
Balanced_accuracy test: 0.2
Balanced_accuracy train: 0.2
              precision    recall  f1-score   support

           1       0.22      1.00      0.37       805
           2       0.00      0.00      0.00       596
           3       0.00      0.00      0.00       720
           4       0.00      0.00      0.00       767
           5       0.00      0.00      0.00       712

    accuracy                           0.22      3600
   macro avg       0.04      0.20      0.07      3600
weighted avg       0.05      0.22      0.08      3600
```

Figura 9. Resultados evaluación Random Forest

```
accuracy test: 0.5394444444444444
accuracy train: 0.9857142857142858
Balanced_accuracy test: 0.5291429429976007
Balanced_accuracy train: 0.9858341723492486
              precision    recall  f1-score   support

           1       0.51      0.64      0.57       805
           2       0.53      0.31      0.39       596
           3       0.64      0.58      0.61       720
           4       0.56      0.54      0.55       767
           5       0.48      0.57      0.52       712

    accuracy                           0.54      3600
   macro avg       0.54      0.53      0.53      3600
weighted avg       0.54      0.54      0.53      3600
```

Figura 10. Resultados evaluación XG Boost

```
accuracy test: 0.5480555555555555
accuracy train: 0.9857142857142858
Balanced_accuracy test: 0.5393792084407047
Balanced_accuracy train: 0.9857190677726615
              precision    recall  f1-score   support

           1       0.54      0.61      0.57       805
           2       0.48      0.34      0.40       596
           3       0.63      0.58      0.61       720
           4       0.60      0.57      0.58       767
           5       0.49      0.60      0.54       712

    accuracy                           0.55      3600
   macro avg       0.55      0.54      0.54      3600
weighted avg       0.55      0.55      0.55      3600
```

La siguiente tabla muestra un resumen de las precisiones obtenidas con cada técnica de aprendizaje y el valor de validación cruzada.

Tabla 3. Resumen de resultados

| Técnica | Balanced accuracy | Validación cruzada |
|---|---|---|
| DecisionTree | 0.4231 | 0.4210 |
| SkLearn | 0.2 | 0.2 |
| Random Forest | 0.5291 | 0.5280 |
| **XG Boost** | **0.5393** | **0.5294** |

De acuerdo con la información de la tabla de resultados, el modelo generado con **XG Boost** es el que obtiene mejor desempeño sobre el dataset de entrenamiento. Los parámetros del modelo ganador se presentan en la siguiente imagen.

Figura 11. Principales parámetros XG Boost

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
              eta=0.03, eval_metric='mlogloss', gamma=0.03, gpu_id=-1,
              importance_type=None, interaction_constraints='',
              learning_rate=0.0299999993, max_delta_step=0, max_depth=20,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=1000, n_jobs=8, num_parallel_tree=1,
              objective='multi:softprob', predictor='auto', random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=2.682472778363189,
              subsample=1, tree_method='exact', validate_parameters=1, ...)
```

## PRUEBAS

Para aplicar el modelo ganador (XG Boost) sobre el dataset de clasificación se requiere que el número y el nombre de las variables para este set de datos sea idéntico al utilizado en el dataset de entrenamiento. Se realiza la predicción y se genera un dataset final conformado por el identificador del inmueble y la clase resultado de este proceso.

Figura 12. Dataset final

| | id | prediccion |
|---|---|---|
| **0** | 1379 | 4 |
| **1** | 691 | 4 |
| **2** | 5425 | 5 |
| **3** | 5687 | 4 |
| **4** | 2295 | 1 |
| **...** | ... | ... |
| **2995** | 14089 | 1 |
| **2996** | 4520 | 1 |
| **2997** | 2008 | 2 |
| **2998** | 1066 | 4 |
| **2999** | 5184 | 3 |

3000 rows × 2 columns

Este conjunto de datos es comparado con la solución planteada por el docente donde se obtiene que la clasificación generada por el modelo tiene un valor de *balanced accuracy* y *f1-score* de 0.53.

El material adicional que incluye notebooks, scripts de limpieza, datasets de consulta, para llevar a cabo todo el proceso está disponible en el repositorio[31].

# CONCLUSIONES

- La aplicación de técnicas de aprendizaje automático permitió solucionar el problema planteado en el caso de estudio, donde se buscaba predecir el precio de venta del metro cuadrado de un inmueble en la ciudad de Buenos Aires.

- Un proceso minucioso de limpieza y preparación de los datos, que incluya el aumento en el número de atributos, transformación de variables categóricas, limpieza de valores nulos o insignificantes, corrección de valores atípicos y formato de datos garantizará un mejor performance de los modelos generados.

- El modelo generado mediante XG Boost fue el mejor calificado con el set de datos de entrenamiento debido a que obtuvo el valor más alto para *balanced accuracy* y validación cruzada.

- El uso de técnicas de minería de texto permiten descubrir relaciones y tendencias ocultas en las palabras sin necesidad de conocer los términos precisos.

# REFERENCIAS

[1] The GeoNames geographical database. Recuperado de: https://www.geonames.org/

[2] Buenos Aires Data. Recuperado de: https://data.buenosaires.gob.ar/dataset/

[3] Repositorio GitHub del proyecto: https://github.com/wilmerescobarb/machine-learning-inmuebles