# Predicción de precios de vivienda

Nikolas Sebastián Rodriguez Alfonso

Científico de Datos

https://github.com/NikolasRodriguezA/Predicciones_Vivienda

# Exploración de Datos

- Dataset con 222 variables y 11571 observaciones, destacando la variable valor_total_avaluo, siendo esta la variable que generará el corpus para un posterior modelamiento.

```
df.shape
```

```
(11571, 222)
```

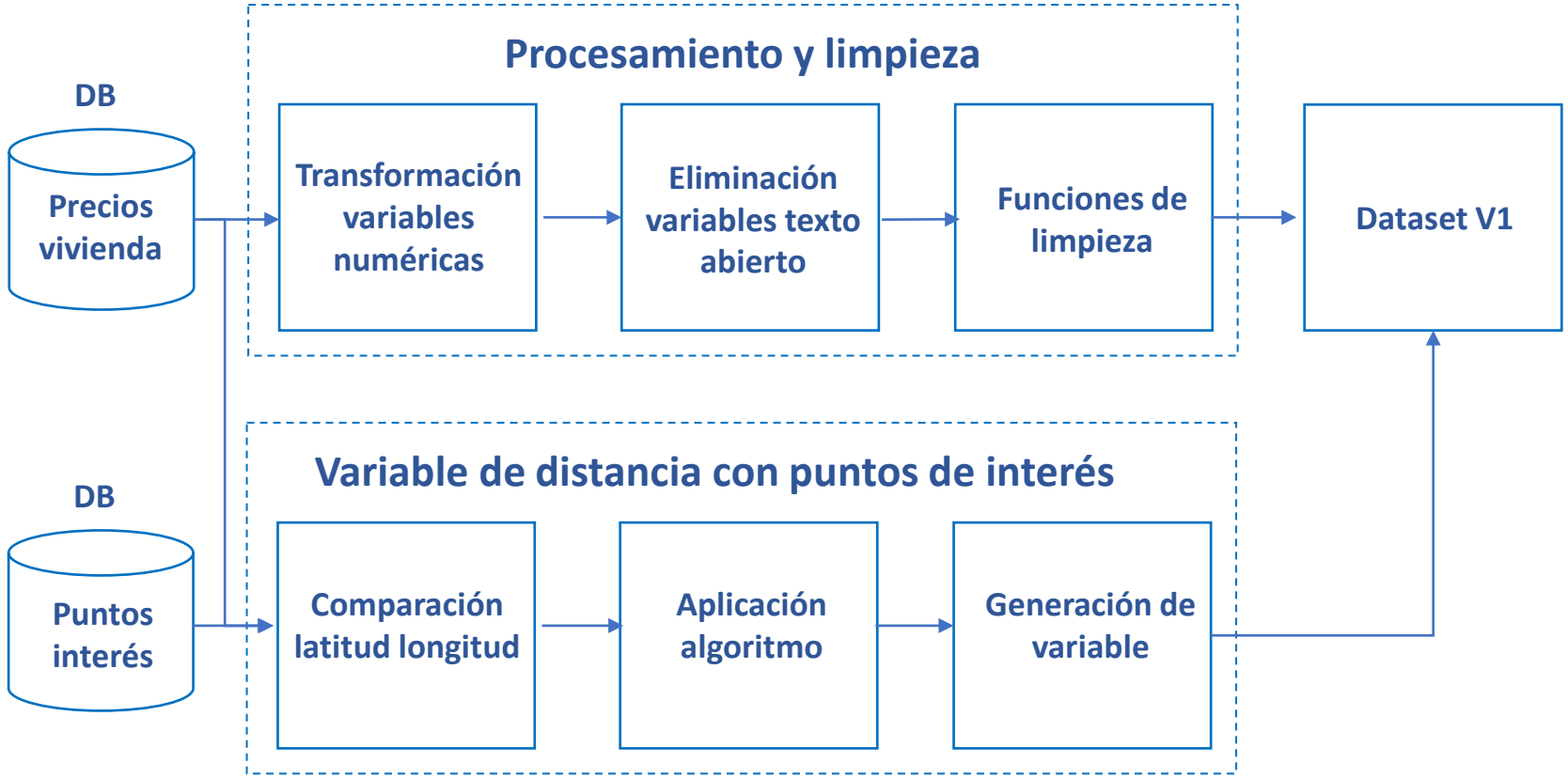
```
df.info()
```

```
RangeIndex: 11571 entries, 0 to 11570
Columns: 222 entries, Unnamed: 0 to Latitud
dtypes: float64(1), int64(18), object(203)
memory usage: 19.6+ MB
```

| :onstruccion | area_otros | area_libre | valor_area_libre | valor_total_avaluo | valor_uvr |
|---|---|---|---|---|---|
| 81968750.0 | 0 | 0 | 0.0 | 145318750,00 | 2522304 |
| 0.0 | 157 | No | 0.0 | 713986654 | 257.23250000000002 |
| 0.0 | 0 | Si | 0.0 | 270500000 | 259.4264 |
| 0.0 | 0 | No | 0.0 | 84840000,00 | 252245 |
| 69306400.0 | 0 | 0 | 0.0 | 96346400 | 259.72770000000003 |
| ... | ... | ... | ... | ... | ... |
| 0 | 0 | No | 0 | 709028000 | 259.57420000000002 |
| 0 | 0 | Si | 2193300 | 158356260 | 254.18279999999999 |
| 0 | 0 | No | 0 | 572610000 | 254.4109 |
| 97997000,00 | 0 | 0 | 0 | 183290000,00 | 2517095 |
| 84500000 | 0 | 0 | 0 | 154500000 | 257.17070000000000 |

# Exploración de Datos

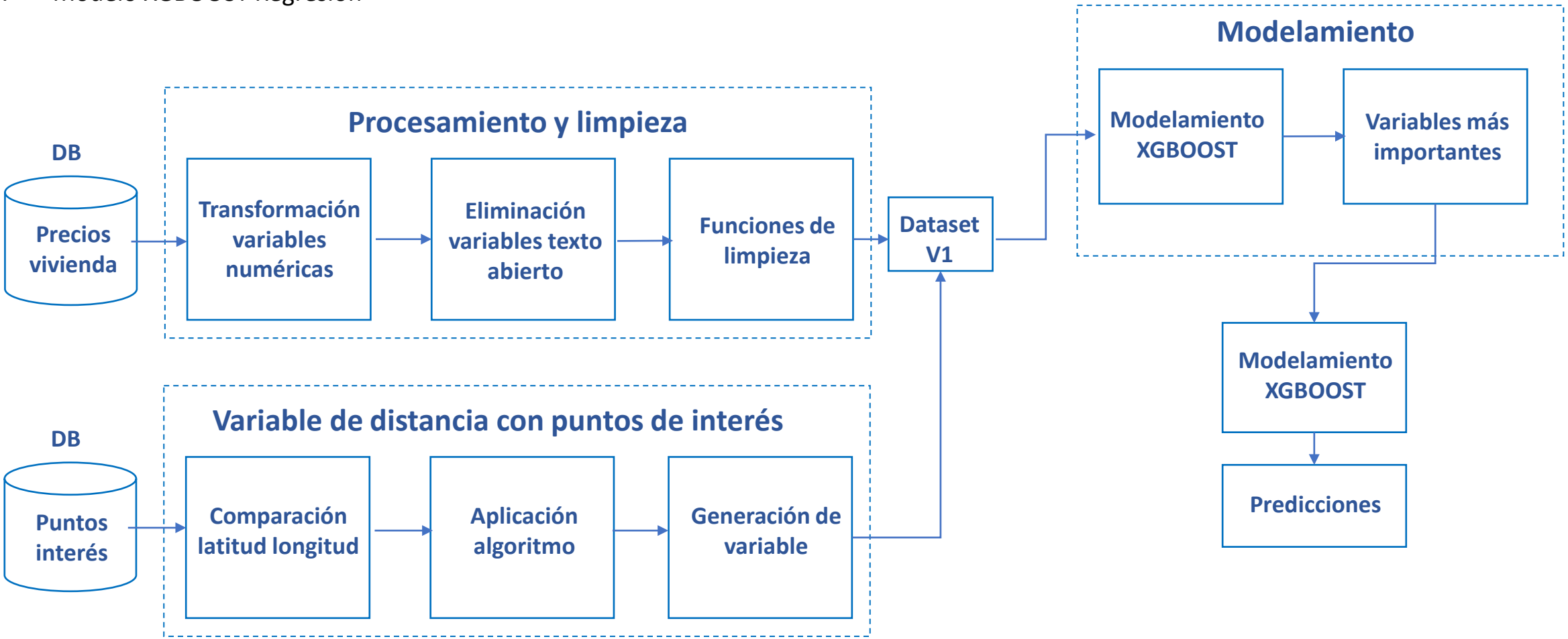
- Se puede observar que el conjunto de datos cuenta con variables numéricas que vienen en formato texto
- Se debe realizar una función de limpieza de datos
- Tras realizar una exploración del dataset pueden observar variables de tipo texto abierto que se eliminan resultando el siguiente dataset



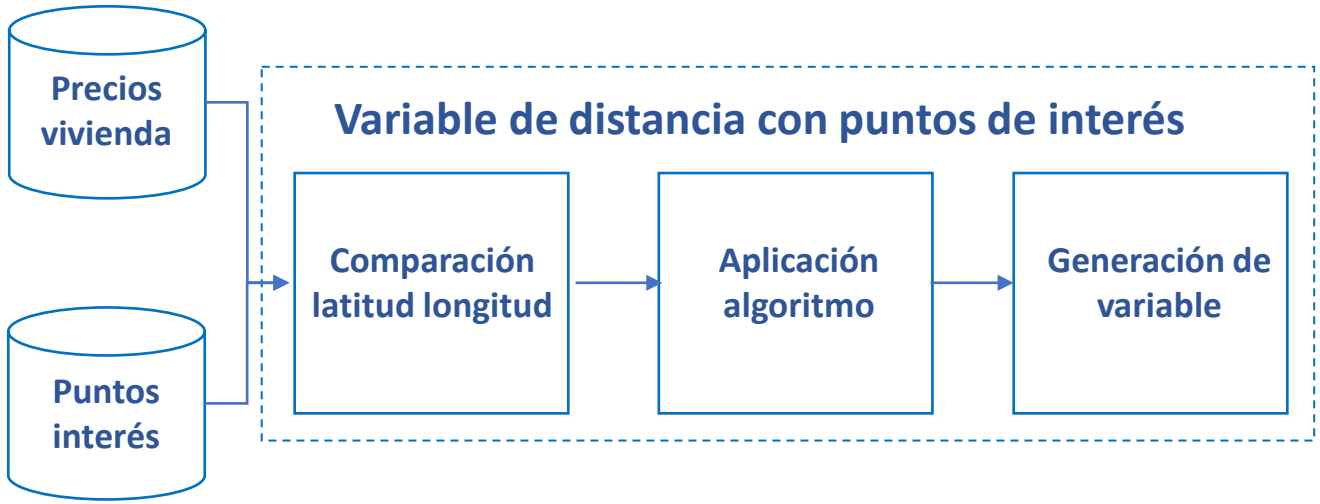
# Técnicas implementadas

Flujo de modelamiento:

1. Limpieza de variables
2. Algoritmo distancias a puntos de interés
3. Modelo XGBOOST (Selección variables)
4. Modelo XGBOOST Regresión



# Técnicas implementadas




## Algoritmo para calcular la menor distancia en kilometros para cada inmueble con los puntos de interes

Obteniendo las distancias de cada coordenada con los puntos de interes dados utilizando el radio de la tierra y la transformación:

$$R = radiodelatierra$$

$$dif_{lat} = lat1 - lat2$$

$$dif_{long} = long1 - long2$$

$$a = sin(\frac{dif_{lat}}{2})^2 + cos(lat1) * sin(\frac{dif_{long}}{2})^2$$

$$C = 2 * arctan(\frac{\sqrt{a}}{\sqrt{1-a}})$$

$$d = R * c$$

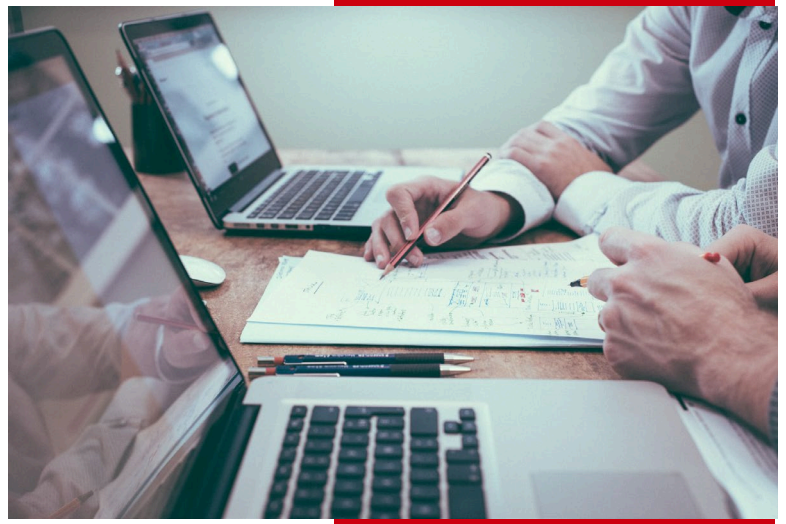
$$d = distanciaenkilometros$$

## Dificultades encontradas

- Para mejoras en el modelamiento se puede dedicar más esfuerzo a la exploración y limpieza de datos.
- Debido al tiempo y a la premura de la solución no se realiza un análisis tan detallado en limpieza y extracción de características, para compensar esto se realiza un modelamiento con bastante poder de aprendizaje como lo es el algoritmo XGBOOST.
- Sin embargo, se logra hacer una limpieza de estos valores para poder ser convertidos a numéricos, debido a que los datos vienen con algunos problemas de formato.

# Aspectos importantes

- Para la limpieza de los datos se realizan dos funciones en donde la primera se realiza una conversión con encapsulación de errores para evitar que el flujo de limpieza se detenga por errores.
- Se realiza una exploración del dataset mirando aquellas variables que no puedan aportar poder predictivo al modelo y puedan ser descartadas (variables texto abierto).
- Puede considerarse este desarrollo como una PoC en donde se pueden encontrar bastas oportunidades de mejora sobre todo en el ámbito de limpieza y exploración de datos.



# Resultados FastText - DBSCAN

## Modelo XGBOOST y busqueda de hiperparámetros con Gridsearch

```
params1 = {
    'n_estimators':[500],
    'max_depth':[3,5],
    'learning_rate':np.linspace(0.001,0.1,2),
    'gamma':np.linspace(0,1,2),#
    'min_child_weight':np.linspace(1,5,2),#
    'subsample':np.linspace(0.3,0.9,2), #
    'colsample_bytree':np.linspace(0.3,0.9,2),#
}


xgb_estimator1 = XGBRegressor(seed=42)

xgb_estimator2 = XGBRegressor(learning_rate =0.1, n_estimators=1000,
      subsample=0.6, colsample_bytree=0.6,seed=42)

gsearch1 = GridSearchCV(estimator=xgb_estimator1,param_grid=params1,
                        scoring='neg_mean_absolute_percentage_error',n_jobs=-1,
                        refit='neg_mean_absolute_percentage_error',cv=4,verbose=11)

xgb_estimator2.fit(X_train,y_train)
```

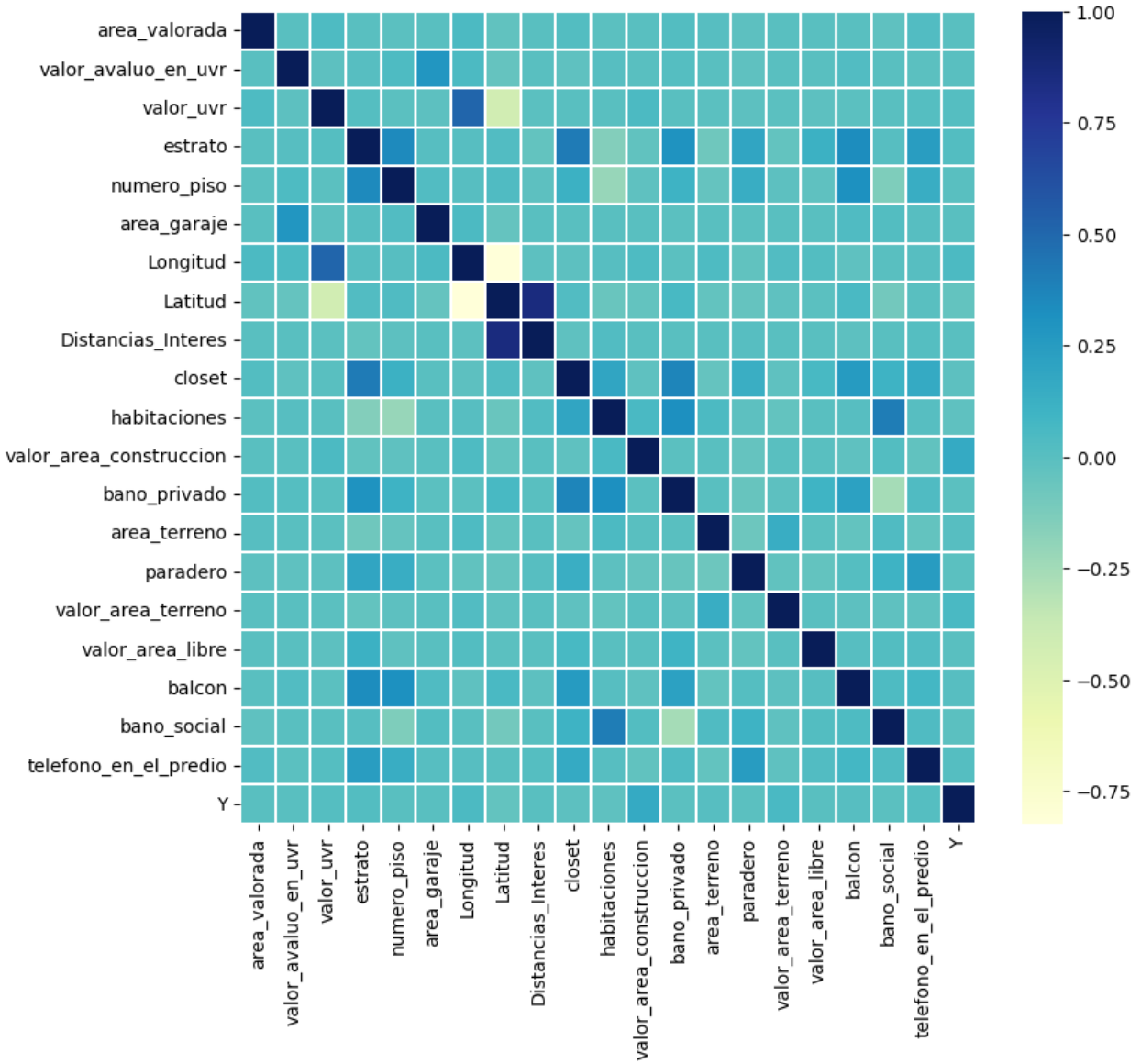
```
XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=0.6,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, feature_types=None, gamma=0, gpu_id=-1,
             grow_policy='depthwise', importance_type=None,
             interaction_constraints='', learning_rate=0.1, max_bin=256,
             max_cat_threshold=64, max_cat_to_onehot=4, max_delta_step=0,
             max_depth=6, max_leaves=0, min_child_weight=1, missing=nan,
             monotone_constraints='()', n_estimators=1000, n_jobs=0,
             num_parallel_tree=1, predictor='auto', random_state=42, ...)
```

- Resultados con la métrica MAPE

| Modelo | Error MAPE |
|---|---|
| XGBOOST sin gridsearch | 7,62 |
| XGBOOST con gridsearch | 50,36 |
| XGBOOST con mejores 20 variables | 6,64 |

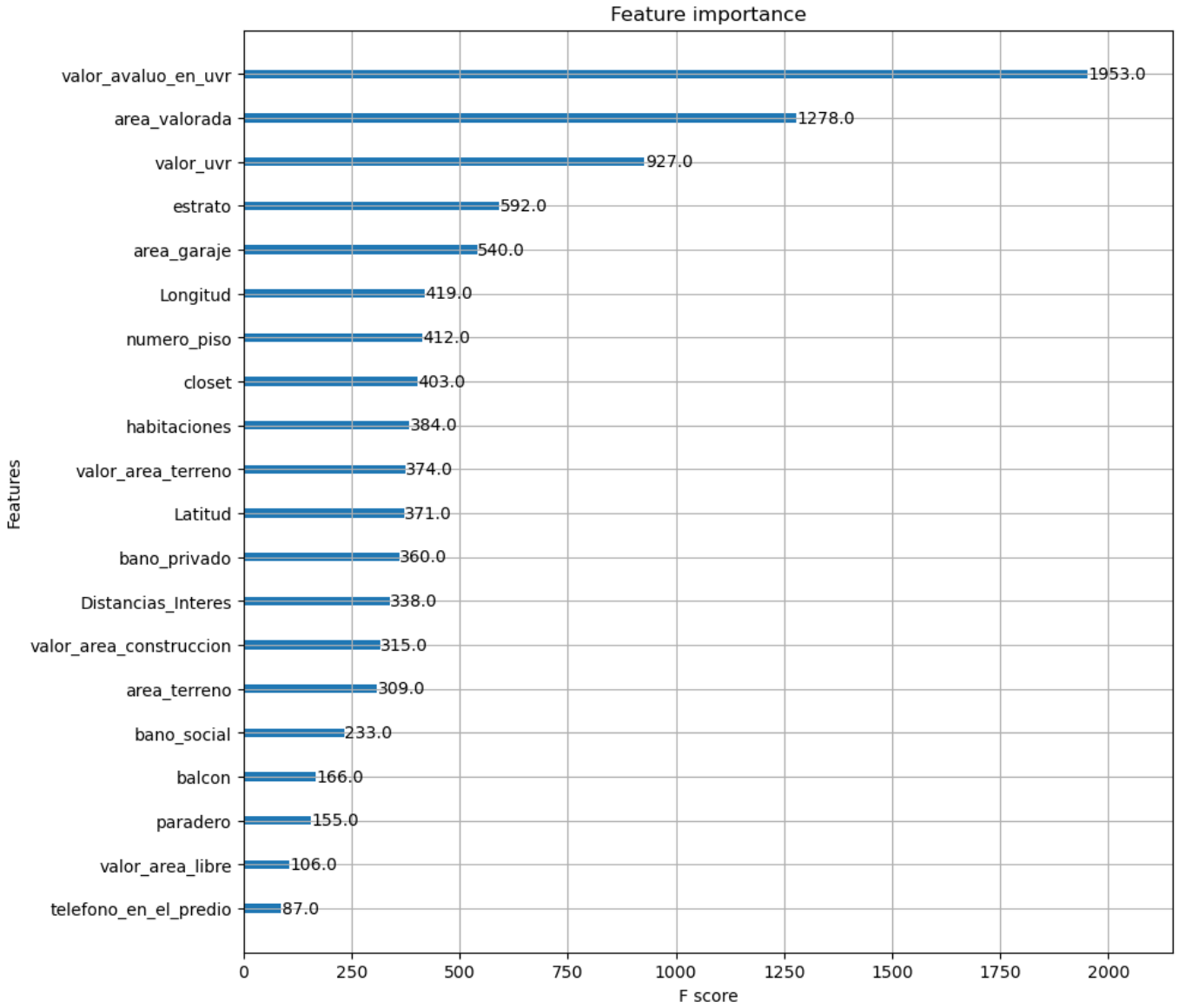
# Resultados Modelado LDA

- Correlación de las 20 variables con más poder predictivo según el modelo XGBOOST:



# Resultados Modelado LDA

- Visualización de las 20 variables con más poder predictivo según el modelo XGBOOST:



# Costos ejecución

- Para la ejecución de ambos modelos se utiliza un equipo con 16 GB de RAM, procesador de 8 núcleos a 3,59 GHz tomando un tiempo de entrenamiento por cada modelo aproximado de medio minuto cada uno.

- No se utiliza ninguna API paga ni hardware en nube.

- Se utiliza la librería de XGBOOST.

- Se realiza la comparación de los puntos coordenados con solo las categorías descritas en el notebook, utilizando la librería tqdm que permite estimar tiempos de ejecución, al realizar la comparación de todos los puntos de interés tardaría alrededor de 6 horas con el hardware descrito anteriormente

# Detalles adicionales y conclusiones

- Se logra desarrollar el modelo con la capacidad de realizar predicciones bajo un error MAPE de 6,64.
- Se logran determinar las 20 variables más influyentes para mejorar las predicciones y por ende las métricas de rendimiento.
- Se logra observar que la variable creada por el algoritmo de distancias entra dentro de las 20 variables más influyentes.

# Predecir tópico del tweet