Trabajo práctico nro. 2. Punto 5.

- Documentación

A lo largo de este trabajo práctico se analiza la base de la competencia Kaggle en la cual se estiman los precios de ventas de propiedades en Melbourne, y el conjunto de datos de scrapings del sitio Airbnb realizado por Tyler Xie, también de una competencia de Kaggle.

El dataset de Melbourne tiene 13580 registros y está conformado por las siguientes variables:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13580 entries, 0 to 13579
Data columns (total 21 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Suburb         13580 non-null  object
 1   Address        13580 non-null  object
 2   Rooms          13580 non-null  int64
 3   Type           13580 non-null  object
 4   Price          13580 non-null  float64
 5   Method         13580 non-null  object
 6   SellerG        13580 non-null  object
 7   Date           13580 non-null  object
 8   Distance       13580 non-null  float64
 9   Postcode       13580 non-null  float64
 10  Bedroom2       13580 non-null  float64
 11  Bathroom       13580 non-null  float64
 12  Car            13518 non-null  float64
 13  Landsize       13580 non-null  float64
 14  BuildingArea   7130 non-null   float64
 15  YearBuilt      8205 non-null   float64
 16  CouncilArea    12211 non-null  object
 17  Lattitude      13580 non-null  float64
 18  Longtitude     13580 non-null  float64
 19  Regionname     13580 non-null  object
 20  Propertycount  13580 non-null  float64
dtypes: float64(12), int64(1), object(8)
memory usage: 2.2+ MB
```

El dataset de Airbnb tiene 22895 registros y las siguientes variables:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 22895 entries, 0 to 22894
Data columns (total 84 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   id                              22895 non-null  int64
 1   listing_url                     22895 non-null  object
 2   scrape_id                       22895 non-null  float64
 3   last_scraped                    22895 non-null  object
 4   name                            22892 non-null  object
 5   summary                         22199 non-null  object
 6   space                           16844 non-null  object
 7   description                     22563 non-null  object
 8   neighborhood_overview           14424 non-null  object
 9   notes                           11546 non-null  object
 10  transit                         14943 non-null  object
 11  access                          15168 non-null  object
 12  interaction                     14537 non-null  object
 13  house_rules                     15032 non-null  object
 14  picture_url                     22895 non-null  object
 15  host_id                         22895 non-null  int64
 16  host_url                        22895 non-null  object
 17  host_name                       22892 non-null  object
 18  host_since                      22892 non-null  object
 19  host_location                   22869 non-null  object
 20  host_about                      13730 non-null  object
 21  host_response_time              15589 non-null  object
 22  host_response_rate              15589 non-null  object
 23  host is superhost               22892 non-null  object
 24  host_thumbnail_url              22892 non-null  object
 25  host_picture_url                22892 non-null  object
 26  host_neighborhood               14927 non-null  object
 27  host_verifications              22895 non-null  object
 28  host_has_profile_pic            22892 non-null  object
 29  host_identity_verified          22892 non-null  object
 30  street                          22895 non-null  object
 31  neighborhood                    17002 non-null  object
 32  city                            22895 non-null  object
 33  suburb                          22872 non-null  object
 34  state                           22834 non-null  object
 35  zipcode                         22749 non-null  float64
 36  smart_location                  22895 non-null  object
 37  country_code                    22895 non-null  object
 38  country                         22895 non-null  object
 39  latitude                        22895 non-null  float64
 40  longitude                       22895 non-null  float64
 41  is_location_exact               22895 non-null  object
 42  property_type                   22895 non-null  object
 43  room_type                       22895 non-null  object
 44  accommodates                    22895 non-null  int64
 45  bathrooms                       22878 non-null  float64
 46  bedrooms                        22890 non-null  float64
 47  beds                            22861 non-null  float64
 48  bed_type                        22895 non-null  object
 49  amenities                       22895 non-null  object
 50  price                           22895 non-null  int64
 51  weekly_price                    2524 non-null   float64
 52  monthly_price                   1891 non-null   float64
 53  security_deposit                15401 non-null  float64
 54  cleaning_fee                    17249 non-null  float64
 55  guests_included                 22895 non-null  int64
 56  extra_people                    22895 non-null  int64
 57  minimum_nights                  22895 non-null  int64
 58  maximum_nights                  22895 non-null  int64
 59  calendar_updated                22895 non-null  object
 60  has_availability                22895 non-null  object
 61  availability_30                 22895 non-null  int64
 62  availability_60                 22895 non-null  int64
 63  availability_90                 22895 non-null  int64
 64  availability_365                22895 non-null  int64
 65  calendar_last_scraped           22895 non-null  object
 66  number_of_reviews               22895 non-null  int64
 67  first_review                    17653 non-null  object
 68  last_review                     17653 non-null  object
 69  review_scores_rating            17189 non-null  float64
 70  review_scores_accuracy          17175 non-null  float64
```

```
 71  review_scores_cleanliness         17182 non-null  float64
 72  review_scores_checkin             17161 non-null  float64
 73  review_scores_communication       17177 non-null  float64
 74  review_scores_location            17162 non-null  float64
 75  review_scores_value               17160 non-null  float64
 76  requires_license                  22895 non-null  object
 77  license                           21 non-null     object
 78  instant_bookable                  22895 non-null  object
 79  cancellation_policy               22895 non-null  object
 80  require_guest_profile_picture     22895 non-null  object
 81  require_guest_phone_verification  22895 non-null  object
 82  calculated_host_listings_count    22895 non-null  int64
 83  reviews_per_month                 17653 non-null  float64
dtypes: float64(19), int64(14), object(51)
memory usage: 14.7+ MB
```

En primer lugar se convierte en base de datos a ambos datasets con el paquete de Python, sqlalchemy

Luego se realizan dos consultas de sql a ambas bases:

- Registro por ciudad: en el caso de la base de Melbourne se hizo una distribución de la variable CouncilArea mientras que en la base de Airbnb se realizó por la variable 'city'
- Registros por ciudad y barrio: en el caso de Melbourne se hizo una distribución por CouncilArea y Suburb mientras que en Airbnb por city y suburb.

A fines de agregar variables de interés a la base de Melbourne, se calculó el precio medio por código postal, la cantidad de viviendas en la base de Melbourne por código postal y los precios promedio por código postal del precio mensual y el precio semanal y luego se hizo un merge con estas nuevas variables y la base de Melbourne.

Dado que se observó que la variable CouncilArea de Melbourne tiene varios missings, se completó la base con las variables city y suburb de Airbnb con un merge. Una vez agregadas dichas variables se convierte la base de datos obtenida en un dataframe para empezar a trabajar con el paquete Pandas de Python. Luego, se eligen las variables de interés para analizar los determinantes del precio de las viviendas de las bases Melbourne.

Las variables que se habían creado a partir de Airbnb: precio medio semanal y precio medio mensual por código postal se descartan porque no sería una referencia válida dada la cantidad de missings que tenían las variables originales.

El dataset obtenido es el siguiente:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13580 entries, 0 to 13579
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   CouncilArea           12211 non-null  object
 1   Suburb                13580 non-null  object
 2   Rooms                 13580 non-null  int64
 3   Bedroom2              13580 non-null  float64
 4   Bathroom              13580 non-null  float64
 5   Car                   13518 non-null  float64
 6   Type                  13580 non-null  object
 7   Price                 13580 non-null  float64
 8   Landsize              13580 non-null  float64
 9   BuildingArea          7130 non-null   float64
 10  YearBuilt             8205 non-null   float64
 11  Postcode              13580 non-null  float64
 12  city                  13456 non-null  object
 13  suburb:1              13456 non-null  object
 14  airbnb_price_mean_zp  13560 non-null  float64
 15  conteo_zipcode        13560 non-null  float64
```

Luego se procede a analizar cada variable:

- En el caso de Price se quitan los outliers con la metodología de IQR
- Se quitan las propiedades sin baño, y con más de 1 habitación
- Las variables BuildingArea y Yearbuilt se dejan igual ya que luego se van a imputar
- En el caso de la variable Landsize se reemplazaron los valores nulos por NAN.
- Se removió un registro con una casa de construcción muy antigua (año 1196)

Luego, como se pedía seleccionar aquellos registros cuyos zipcodes tuvieron asociado una mínima cantidad de viviendas, se analizó mediante histograma y boxplot y se eli gieron aquellos zipcode con 20 viviendas o más.

El dataset final está conformado de la siguiente manera:

```
df_final.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11477 entries, 0 to 13579
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Bathroom              11477 non-null  float64
 1   Bedroom2              11477 non-null  float64
 2   BuildingArea          6006 non-null   float64
 3   Car                   11477 non-null  float64
 4   CouncilArea           10433 non-null  object
 5   Landsize2             9593 non-null   float64
 6   Postcode              11477 non-null  float64
 7   Price                 11477 non-null  float64
 8   Rooms                 11477 non-null  int64
 9   Suburb                11477 non-null  object
 10  Type                  11477 non-null  object
 11  YearBuilt             6945 non-null   float64
 12  airbnb_price_mean_zp  11477 non-null  float64
 13  city                  11457 non-null  object
 14  conteo_zipcode        11477 non-null  float64
 15  suburb:1              11457 non-null  object
dtypes: float64(10), int64(1), object(5)
memory usage: 1.5+ MB
```

La segunda parte del práctico tiene como objetivo imputar los valores faltantes de YearBuilt y BuildingArea y luego aplicar el PCA.

Dado el tratamiento que se efectuó a Landsize (de reemplazar valores nulos por NAS) también se va a imputar.

Para imputar es necesario dicotomizar las variables categóricas. Por esta razón se procede a aplicar OneHotEncoder a las variables categóricas.

El medio de imputación de las variables YearBuilt, BuildingArea y Landsize2 es por el Iterative imputer con el regresor KNN.

Antes y después de imputar las variables, se escalaron con el método StandardScaler a fines de que cada variable tenga media 1 y varianza 0.

Luego se pasó a realizar el PCA con las siguientes variables numéricas:

- 'Bathroom',
- 'Rooms',
- 'airbnb_price_mean_zp',
- 'conteo_zipcode',
- 'YearBuilt',
- 'BuildingArea',
- 'Bedroom2',
- 'Landsize2',
- 'Car'

Y el target era la variable Price.

```
## se seleccionan las variables numéricas para aplicar el PCA
features=df_6[['Bathroom', 'Rooms', 'airbnb_price_mean_zp',
       'conteo_zipcode', 'YearBuilt', 'BuildingArea','Bedroom2',
       'Landsize2','Car']]
```

```
features
```

	Bathroom	Rooms	airbnb_price_mean_zp	conteo_zipcode	YearBuilt	BuildingArea	Bedroom2	Landsize2	Car
0	1.0	2	130.6240	258.0	2002.0	112.2	2.0	202.0	1.0
1	1.0	2	130.6240	258.0	1900.0	79.0	2.0	156.0	0.0
2	2.0	3	130.6240	258.0	1900.0	150.0	3.0	134.0	0.0
3	2.0	3	130.6240	258.0	1971.6	73.8	3.0	94.0	1.0
4	1.0	4	130.6240	258.0	2014.0	142.0	3.0	120.0	2.0
...	...	...	...	...	...	...	...	...	...
11472	2.0	3	92.2857	21.0	1927.6	90.4	3.0	256.0	2.0
11473	2.0	4	124.0265	189.0	1981.0	156.2	4.0	652.0	2.0
11474	2.0	3	191.0946	74.0	1995.0	133.0	3.0	333.0	2.0
11475	2.0	3	191.0946	74.0	1997.0	226.4	3.0	436.0	4.0
11476	1.0	4	135.6098	82.0	1920.0	112.0	4.0	362.0	1.0

11477 rows × 9 columns

```
# Se convierte el array a dataframe para añadir nombres a los ejes.
pd.DataFrame(
    data    = pca.components_,
    columns = features.columns,
    index   = ['PC1', 'PC2', 'PC3', 'PC4','PC5','PC6','PC7','PC8','PC9']
)
```

	Bathroom	Rooms	airbnb_price_mean_zp	conteo_zipcode	YearBuilt	BuildingArea	Bedroom2	Landsize2	Car
PC1	0.415554	0.538826	-0.111089	-0.223304	0.048293	0.256407	0.535504	-0.028462	0.345145
PC2	0.207368	-0.082870	0.119229	0.191162	0.681225	-0.068468	-0.079078	0.628366	0.171814
PC3	0.252388	0.105968	0.732629	0.514992	-0.268022	0.146264	0.105888	-0.007870	-0.136405
PC4	-0.087637	0.093547	-0.522535	0.423788	-0.348125	0.323850	0.089324	0.496677	-0.224035
PC5	-0.131577	-0.202218	0.152385	-0.234452	0.164526	0.889426	-0.207255	-0.049782	0.005160
PC6	-0.254534	0.016784	0.333016	-0.486628	-0.438996	-0.102557	0.019276	0.575956	0.227801
PC7	-0.191814	-0.130351	-0.075584	0.391175	-0.147428	0.040799	-0.135253	-0.144118	0.851482
PC8	-0.770566	0.338605	0.154237	0.154866	0.314602	0.002884	0.369590	-0.051207	-0.074832
PC9	0.016375	-0.714143	-0.005786	-0.001758	-0.008814	0.005189	0.699696	-0.000840	0.003882

Luego de aplicar un

- pca_dummy_features = pca.fit_transform(x)
- Se transforma en dataframe

  principalDf = pd.DataFrame(data = pca_dummy_features)

```
finalDf = pd.concat([principalDf, y], axis = 1)
finalDf
```

	PCA1	PCA2	PCA3	PCA4	PCA5	PCA6	PCA7	PCA8	PC9	Price
0	-1.491180	0.669899	-0.526078	-0.513842	0.455927	-0.827741	-0.160623	0.473323	0.008069	1480000.0
1	-2.099367	-1.737929	0.450100	0.746274	-0.364728	0.375092	-0.602246	-0.472987	0.031080	1035000.0
2	-0.119071	-1.636021	1.176335	1.011793	-0.412743	-0.069419	-1.161357	-0.942097	0.037851	1465000.0
3	0.178406	0.142272	0.310110	-0.274194	-0.669878	-0.773031	-0.609902	-0.301945	0.018098	850000.0
4	0.695554	0.799755	-0.394383	-0.529800	0.124807	-0.756761	0.294281	1.635643	-0.777883	1600000.0
...	...	...	...	...	...	...	...	...	...	...
11472	0.784922	-0.819814	-0.413338	0.132658	-0.686267	0.283029	0.240955	-1.074344	0.041235	582000.0
11473	1.966192	0.487021	0.201739	-0.013110	-0.341511	-0.407683	-0.129538	0.393091	0.001710	1245000.0
11474	0.732866	0.924266	0.594133	-1.406560	0.279935	-0.112351	-0.144876	-0.066972	0.012620	1031000.0
11475	1.702186	1.321310	0.412543	-1.588749	1.083982	0.299413	1.701206	-0.208727	0.024924	1170000.0
11476	0.794624	-1.498673	0.388493	0.536079	-0.699476	0.793492	-0.600720	1.073425	-0.014003	1285000.0

11477 rows × 10 columns

```
print("Explained variance ratio")
print(pca.explained_variance_ratio_)
```

```
Explained variance ratio
[0.32230015 0.1338537  0.11862371 0.09945999 0.09505685 0.09486834
 0.07772334 0.05174584 0.00636809]
```

Una vez analizado las componentes, se decide elegir la primera ya que explica el 32% de la variabilidad de los datos.

Por último se agrega la columna de PCA para cada observación del dataset obtenido en el práctico 1 y con las variables ya imputadas.

```
final2=pd.concat([final1,principalDf[principalDf.columns[:1]]],axis=1)
final2
```

	Bathroom	Bedroom2	Car	CouncilArea	Postcode	Price	Rooms	Suburb	Type	airbnb_price_mean_zp	city	conteo_zipcode
0	1.0	2.0	1.0	Yarra	3067.0	1480000.0	2	Abbotsford	h	130.6240	Yarra	258.0
1	1.0	2.0	0.0	Yarra	3067.0	1035000.0	2	Abbotsford	h	130.6240	Yarra	258.0
2	2.0	3.0	0.0	Yarra	3067.0	1465000.0	3	Abbotsford	h	130.6240	Yarra	258.0
3	2.0	3.0	1.0	Yarra	3067.0	850000.0	3	Abbotsford	h	130.6240	Yarra	258.0
4	1.0	3.0	2.0	Yarra	3067.0	1600000.0	4	Abbotsford	h	130.6240	Yarra	258.0
...	...	...	...	...	...	...	...	...	...	...	...	...
11472	2.0	3.0	2.0	NaN	3049.0	582000.0	3	Westmeadows	h	92.2857	Hume	21.0
11473	2.0	4.0	2.0	NaN	3150.0	1245000.0	4	Wheelers Hill	h	124.0265	Monash	189.0
11474	2.0	3.0	2.0	NaN	3016.0	1031000.0	3	Williamstown	h	191.0946	Hobsons Bay	74.0
11475	2.0	3.0	4.0	NaN	3016.0	1170000.0	3	Williamstown	h	191.0946	Hobsons Bay	74.0
11476	1.0	4.0	1.0	NaN	3013.0	1285000.0	4	Yarraville	h	135.6098	Maribyrnong	82.0

11477 rows × 17 columns

suburb:1	Landsize2	YearBuilt	BuildingArea	PCA1
Abbotsford	202.0	2002.0	112.2	-1.491180
Abbotsford	156.0	1900.0	79.0	-2.099367
Abbotsford	134.0	1900.0	150.0	-0.119071
Abbotsford	94.0	1971.6	73.8	0.178406
Abbotsford	120.0	2014.0	142.0	0.695554
...	...	...	...	...
Westmeadows	256.0	1927.6	90.4	0.784922
Wheelers Hill	652.0	1981.0	156.2	1.966192
Williamstown	333.0	1995.0	133.0	0.732866
Williamstown	436.0	1997.0	226.4	1.702186
Yarraville	362.0	1920.0	112.0	0.794624