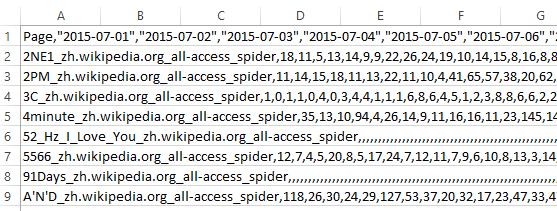
# Relevamiento del pre-procesado de datos

**Tiene los datos así**



Básicamente reemplaza valores  
nan ,,, en el CSV por 0 para cálculos

Luego vuelve a poner los nan devuelta al final

**Después básicamente 4 columnas 1 valor para cada página  
hacen por cada url de wikipedia**

Finds agent types (spider, desktop, mobile, all) for each unique url, i.e. groups pages by agents

**Despues calcula Calcula autocorrelación**

Calcula autocorrelación para  
# Yearly(annual) autocorrelation  
batch de 365  
# Quarterly autocorrelation  
int(round(365.25/4)  
"Percent of undefined autocorr = yearly:%.3f, quarterly:%.3f" % (year\_unknown\_pct, quarter\_unknown\_pct)

**Luego calcula las características de las páginas site, country, agent Las cuales son strings**

Calculates page features (site, country, agent, etc) from urls

pd.DataFrame({  
 **'agent'**: agents,  
 **'site'**: sites,  
 **'country'**: countries,  
 **'term'**: terms,  
 **'marker'**: markers,  
 **'page'**: source  
})

Dropea [**'term'**, **'marker'**]

Aplica one-hot encoding a las características de la página y normaliza  
ya que sus caracteristicas site, country, agent, etc son string

**Calcula Popularidad por página por mediana**

page\_popularity = df.median(axis=1)

Normalizacion:  
page\_popularity = (page\_popularity - page\_popularity.mean()) /page\_popularity.std()

Normaliza tambien las autocorrelaciones yearly y quaterly

Agrega los nans devuelta al dataframe

Queda al final

*# Put NaNs back*df[nans] = np.NaN

*# Assemble final output*tensors = dict(  
 hits=df,#dataframe original  
 lagged\_ix=lagged\_ix,#revisar  
 page\_map=page\_map,# *agent types (spider, desktop, mobile, all) for each unique url*  
 page\_ix=df.index.values,  
 pf\_agent=encoded\_page\_features[**'agent'**],#agente del one hot encoding  
 pf\_country=encoded\_page\_features[**'country'**], #pais del one hot encoding  
 pf\_site=encoded\_page\_features[**'site'**], #sitio del one hot encoding  
 page\_popularity=page\_popularity, #mediana  
 year\_autocorr=year\_autocorr, #autocorrelacion anual  
 quarter\_autocorr=quarter\_autocorr, #autocorrelacion 1 cuarto del año  
 dow=dow,#nose#revisar  
)  
plain = dict(  
 features\_days=len(features\_days),  
 data\_days=len(df.columns),  
 n\_pages=len(df),  
 data\_start=data\_start,  
 data\_end=data\_end,  
 features\_end=features\_end  
  
)