Afectación de la promoción estudiantil secundaria con respecto a datos demográficos y de crímenes en Costa Rica

MARIO ALBERTO BARRANTES QUESADA

Fuentes de datos

Datos de crímenes obtenidos del OIJ

- https://pjenlinea3.poder-judicial.go.cr/estadisticasoij/

Estadísticas demográficas de los distritos

- http://inec.cr/documento/censo-2011-poblacion-total-por-zona-y-sexo-segun-provincia-canton-y-distrito
- https://en.wikipedia.org/wiki/List_of_districts_of_Costa_Rica

Índice de desarrollo distrital

- https://documentos.mideplan.go.cr/share/s/T3CmePFRSdCAUc1q50kZQA

Datos de la promoción estudiantil en escuelas y colegios

- https://www.mep.go.cr/indicadores_edu/autotabulaciones.html

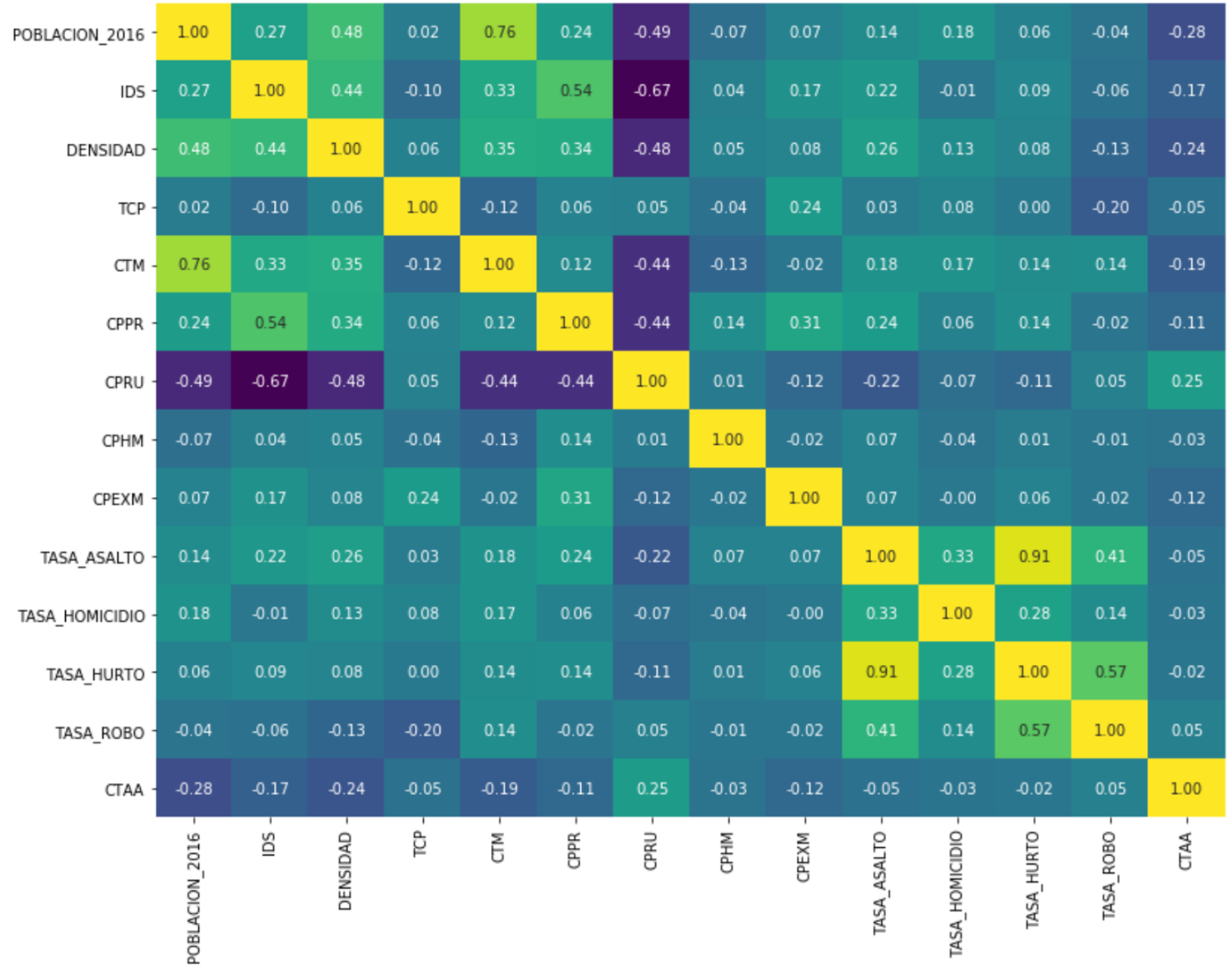
Procesamiento de datos

- Se descargan los datos de crímenes del OIJ mediante llamadas a un web service. Se debe realizar una llamada por cada distrito ya que si descargo toda la información, esta viene incompleta.
- Se descargan manualmente los archivos del IDS (2016) y del Censo del INEC (2011). Se transforman manualmente estos archivos de XSL a CSV para facilidad.
- Se descargan manualmente los archivos del MEP, sobre datos de escuelas, colegios y extranjeros en escuelas y colegios. Se transforman manualmente estos archivos de XSL a CSV para facilidad.
- Se unen todos los datos 8 archivos .csv por el ZIPCODE. Y se genera un set de datos de 482 filas (de los distritos) y 53 columnas.
- Se generan 17 columnas adicionales calculadas por ejemplo: Tasas de Criminalidad (se divide el total de la población entre la cantidad de crímenes), el crecimiento de la población del 2011 al 2016, las proporciones de aprobación de escuelas y colegios, proporciones de escuelas privadas, públicas, rurales, urbanas, proporción de extranjeros en escuelas y colegios, etc.

```
url = 'https://pjenlinea3.poder-judicial.go.cr/estadisticasoij/Home/obtenerDatosDescargas'
headers = {'Content-type': 'application/json', 'Accept': 'text/plain'}
json_template = '{"TN_FechaInicio":20180101,"TN_FechaFinal":20181231,' \
                '"TC_Provincias":"%s","TC_Cantones":"%s","TC_Distritos":"%s",' \
                '"TC_Delito":"1,2,3,4,5,6","TC_Victima":"1,2,3,4,5","TC_Modalidades":"0"}'
crime_cols = ['category', 'sub_category', 'date', 'hour', 'victim_category', 'victim_sub_category', 'age_category',
              'sex', 'nationality', 'province', 'canton', 'district', 'zipcode']

def download_csv(zipcode):
    zipcode = str(zipcode)
    json_object = json_template % (zipcode[0:1], zipcode[0:3], zipcode)
    json_data = {
        'pJson': json_object,
        'pExtension': 'csv'
    }
    response = requests.post(url, headers=headers, json=json_data)
    if response.status_code == 200:
        return response.content
    else:
        print('There was an error: ', response.status_code)
        return None

def parse_csv(csv, row):
    csv = csv.decode("utf-8")
    lines = csv.splitlines()
    crimes = []
    index = 0
    for line in lines:
        index += 1
        if index == 1:
            continue

        cols = line.split(',')
        category = cols[0].strip()
        sub_category = cols[1].strip()
        date = cols[2].strip()
        hour = cols[3].strip()
        victim_category = cols[4]
        victim_sub_category = cols[5]
        age_category = cols[6]
        sex = cols[7]
        nationality = cols[8]
        province = str(row['Province']).upper()
        canton = str(row['Canton']).upper()
        district = str(row['District']).upper()
        zipcode = row['Code']

        crime = [category, sub_category, date, hour, victim_category, victim_sub_category,
                 age_category, sex, nationality, province, canton, district, zipcode]
        crimes.append(crime)

    return crimes


df = pd.read_csv('data/distritos.csv')
all_crimes = pd.DataFrame(columns=crime_cols)
for index, row in df.iterrows():
    print(row['Province'] + ' ' + row['Canton'])
    code = row['Code']
    csv = download_csv(code)
    crimes = parse_csv(csv, row)
    df = pd.DataFrame(crimes, columns=crime_cols)
    all_crimes = all_crimes.append(df, ignore_index=True)

all_crimes.to_csv('data/crimenes.csv', encoding='utf-8-sig')
```

```
def load_csv(spark, path):
    df = spark \
        .read \
        .format('csv') \
        .option('path', path) \
        .option("header", "true") \
        .load()

    return df


def process_extranjeros_escuelas(escuelas_extranjeros_df):
    group = escuelas_extranjeros_df.groupby(['ZIPCODE'])
    df = group.agg({'TOTT': 'sum'}).withColumnRenamed("SUM(TOTT)", "EEXM")
    return df


def process_extranjeros_colegios(colegios_extranjeros_df):
    group = colegios_extranjeros_df.groupby(['ZIPCODE'])
    df = group.agg({'TOTT': 'sum'}).withColumnRenamed("SUM(TOTT)", "CEXM")
    return df


def process_escuelas(escuelas_csv):
    sector_cat = escuelas_csv.select('SECTOR').distinct().rdd.flatMap(lambda x: x).collect()
    zona_cat = escuelas_csv.select('ZONA').distinct().rdd.flatMap(lambda x: x).collect()
    sector_exprs = [funcs.when(funcs.col('SECTOR') == cat, 1).otherwise(0).alias('SECTOR_' + str(cat))
    zona_exprs = [funcs.when(funcs.col('ZONA') == cat, 1).otherwise(0).alias('ZONA_' + str(cat)) for ca

    colegios_csv = escuelas_csv.select(escuelas_csv.columns + sector_exprs + zona_exprs)

    group = colegios_csv.groupby('ZIPCODE')
    df = group.agg({'MFT': 'sum', 'MFH': 'sum', 'MFM': 'sum', 'RET': 'sum', 'REH': 'sum', 'REM': 'sum',
                    'APH': 'sum', 'APM': 'sum', 'SECTOR_1': 'sum', 'SECTOR_2': 'sum', 'SECTOR_3': 'sum
                    'ZONA_1': 'sum', 'ZONA_2': 'sum'})\
        .withColumnRenamed("SUM(MFT)", "ETM")\
        .withColumnRenamed("SUM(MFH)", "EHM") \
        .withColumnRenamed("SUM(MFM)", "EMM") \
        .withColumnRenamed("SUM(RET)", "ETR")\
        .withColumnRenamed("SUM(REH)", "EHR")\
        .withColumnRenamed("SUM(REM)", "EMR")\
        .withColumnRenamed("SUM(APT)", "ETA")\
        .withColumnRenamed("SUM(APH)", "EHA")\
        .withColumnRenamed("SUM(APM)", "EMA")\
        .withColumnRenamed("SUM(SECTOR_1)", "EPU")\
        .withColumnRenamed("SUM(SECTOR_2)", "EPR")\
        .withColumnRenamed("SUM(SECTOR_3)", "ESV")\
        .withColumnRenamed("SUM(ZONA_1)", "EUR")\
        .withColumnRenamed("SUM(ZONA_2)", "ERU")

    df = df.withColumn('ETAA', df['ETA'] / df['ETM'])
    df = df.withColumn('ETAAH', df['EHA'] / df['EHM'])
    df = df.withColumn('ETAAM', df['EMA'] / df['EMM'])

    return df
```

Correlación

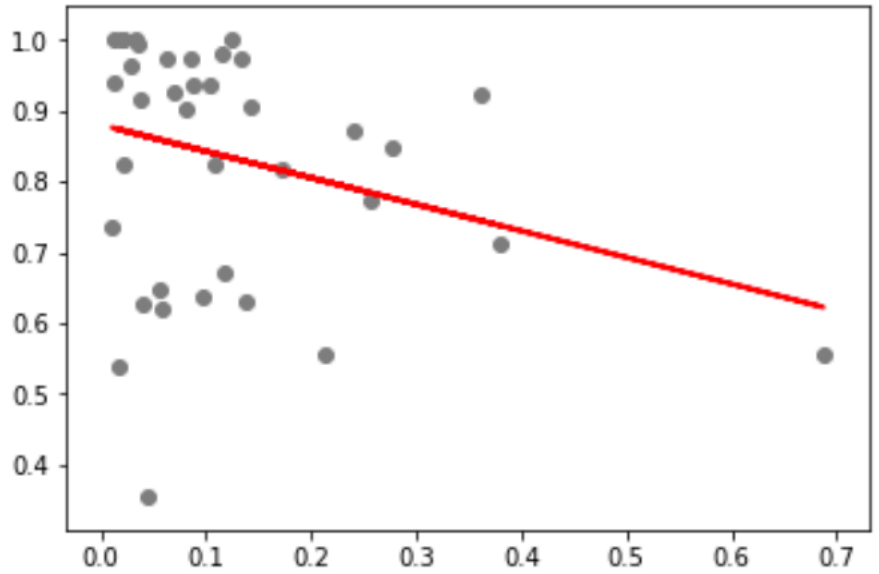


Nomenclatura:

- **IDS**: Índice de Desarrollo Social
- **TCP**: Tasa Crecimiento Población
- **CTM**: Colegio Total Matricula
- **CPPR**: Proporción Colegios Privados
- **CPRU**: Proporción Colegios Rurales
- **CPHM**: Proporción Hombres-Mujeres
- **CPEXM**: Proporción Extranjeros Matriculados
- **CTAA**: Colegio Tasa de Aprobación

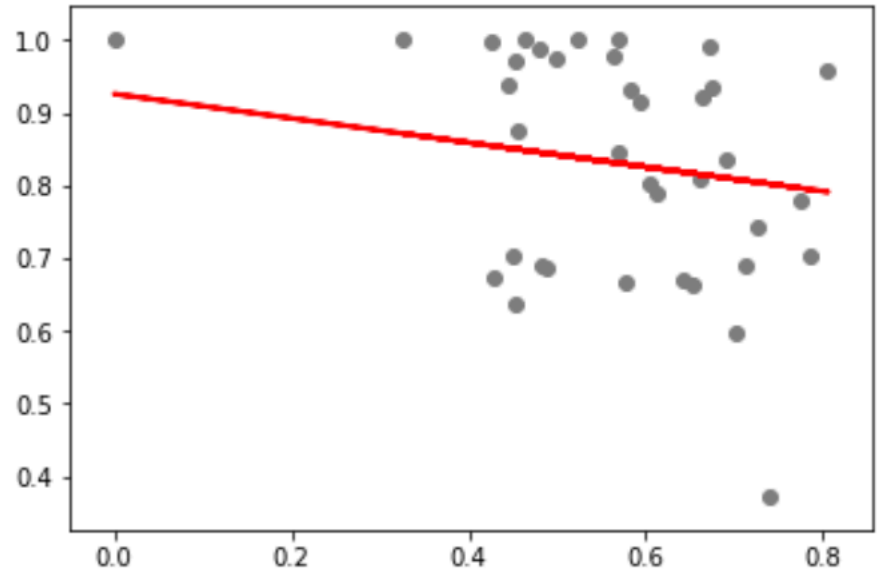
Regresión Lineal
Datos Demográficos

Población 2016



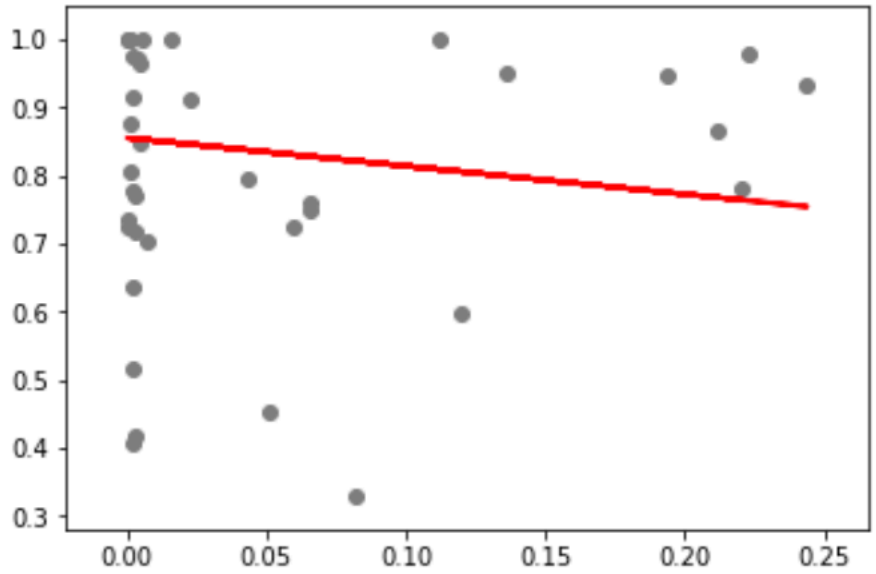
Mean Squared Error: 0.0268050015472651

Índice de Desarrollo Social



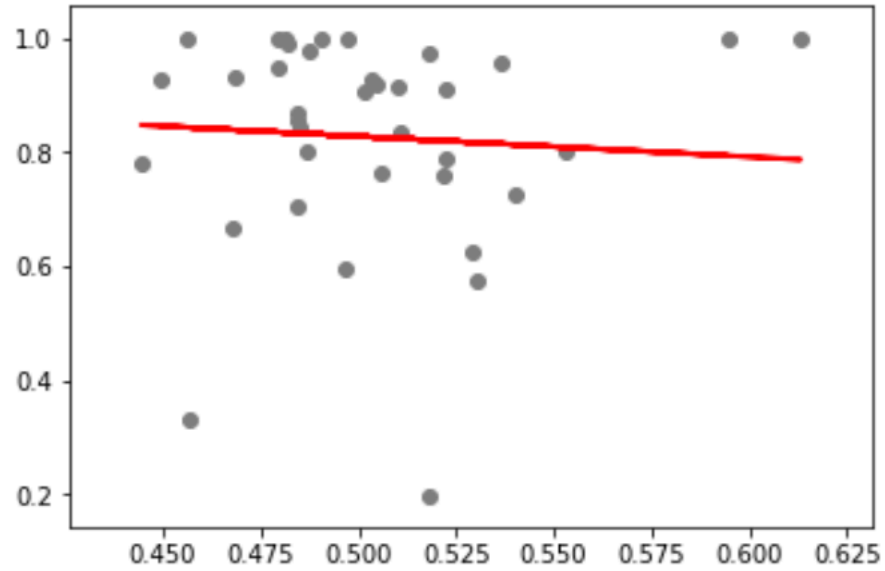
Mean Squared Error: 0.021204439962728038

Densidad



Mean Squared Error: 0.038710832780868 83

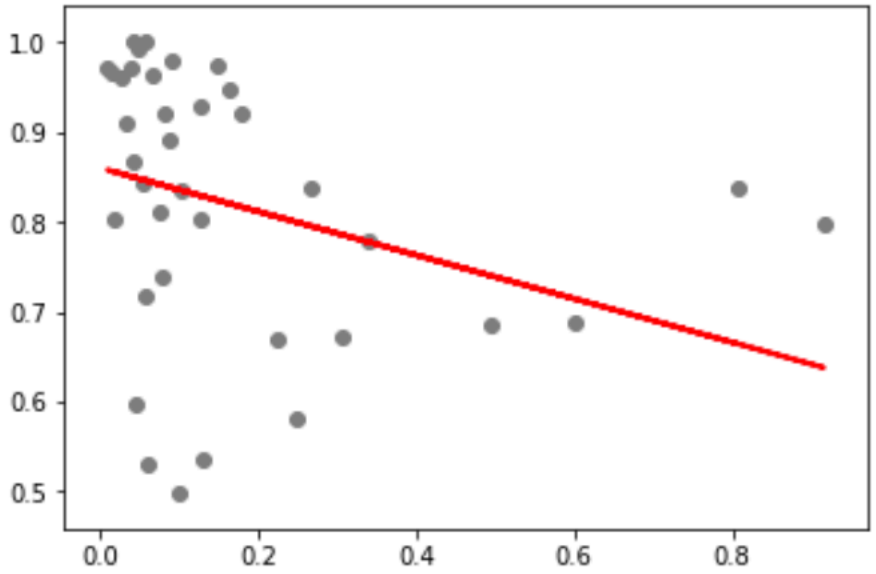
Tasa de Crecimiento de la Población



Mean Squared Error: 0.03386764041597861

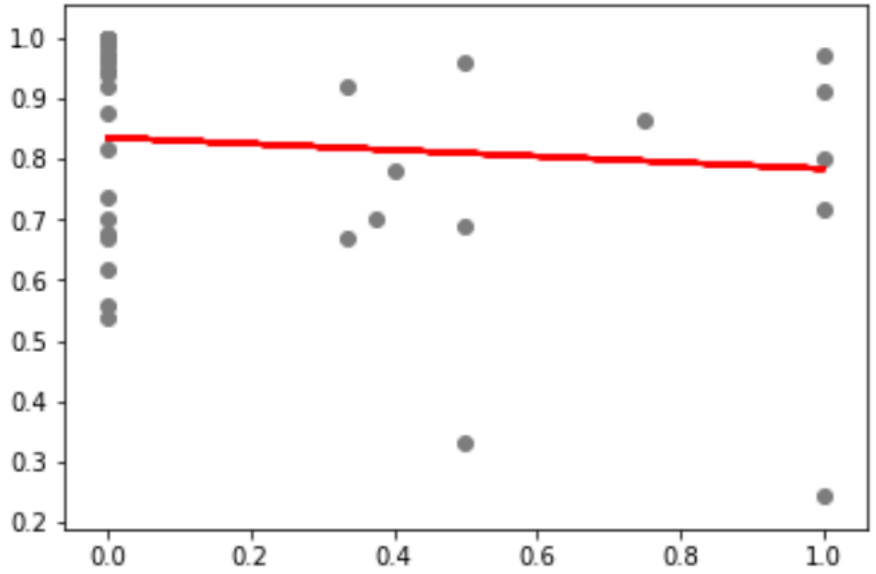
Regresión Lineal
Datos de Colegios

Total Matriculados Colegio

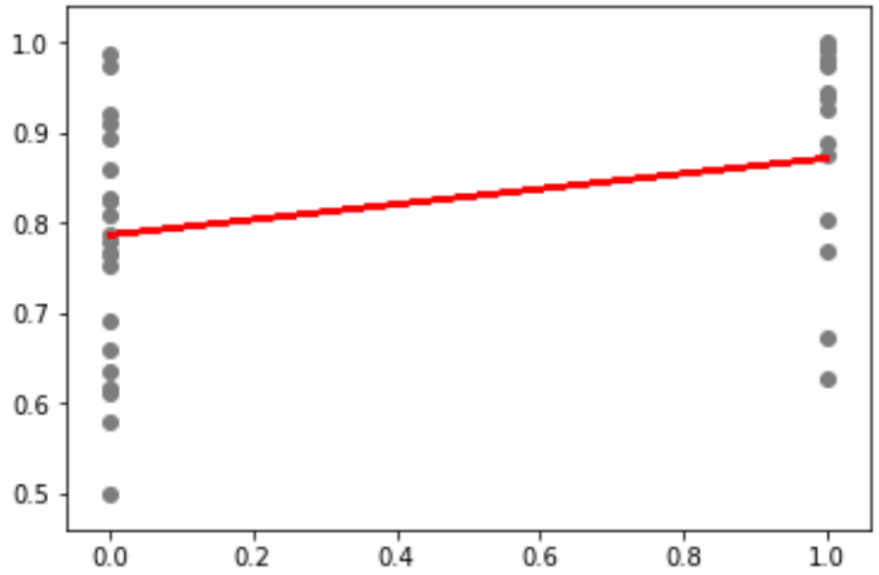


Mean Squared Error: 0.02059255355941803

Proporción de Colegios Privados

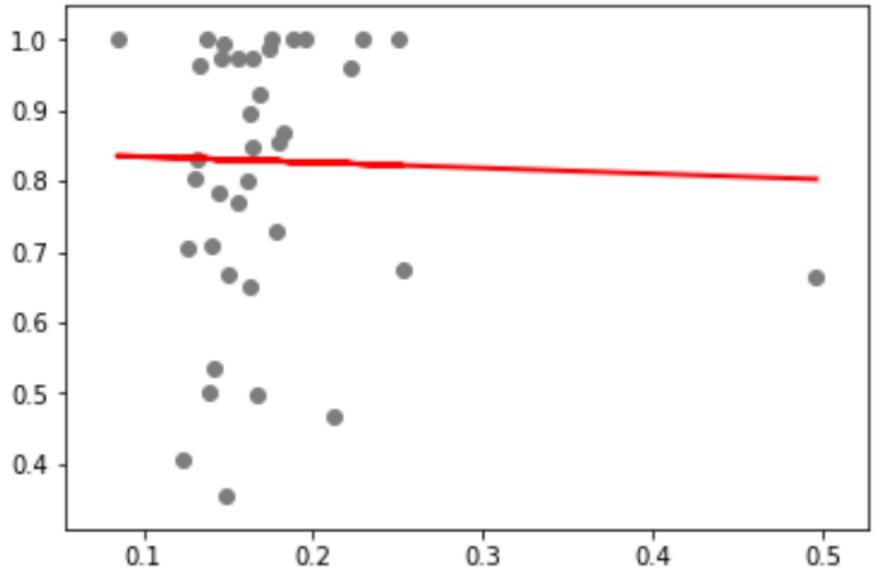


Proporción de Colegios Rurales



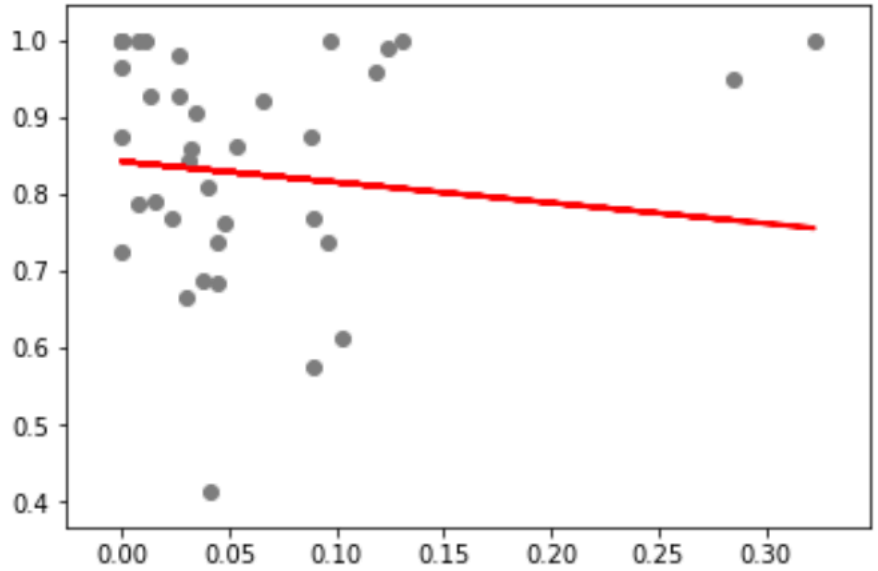
Mean Squared Error: 0.015513582235255 39

Colegios Proporción Hombres/Mujeres



Mean Squared Error: 0.036794117245144 13

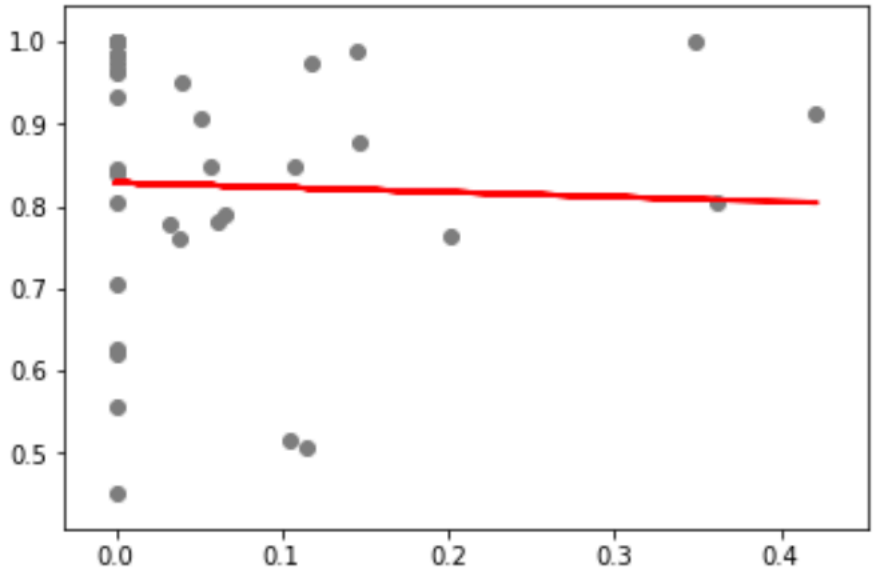
Colegios Proporción Extranjeros Matriculados



Mean Squared Error: 0.021919246362865324

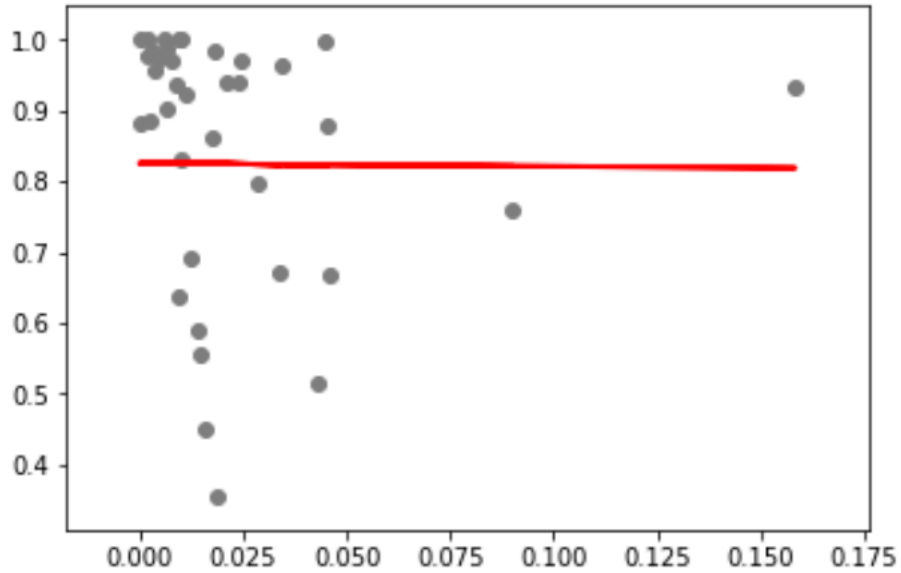
Regresión Lineal
Datos de Crímenes

Tasa de Homicidios



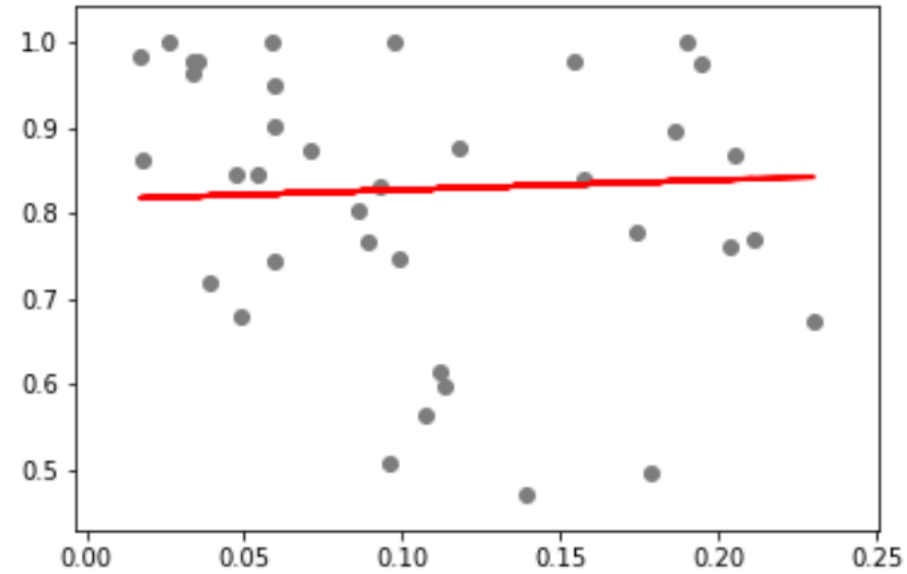
Mean Squared Error: 0.02548757571845969

Tasa de Hurtos



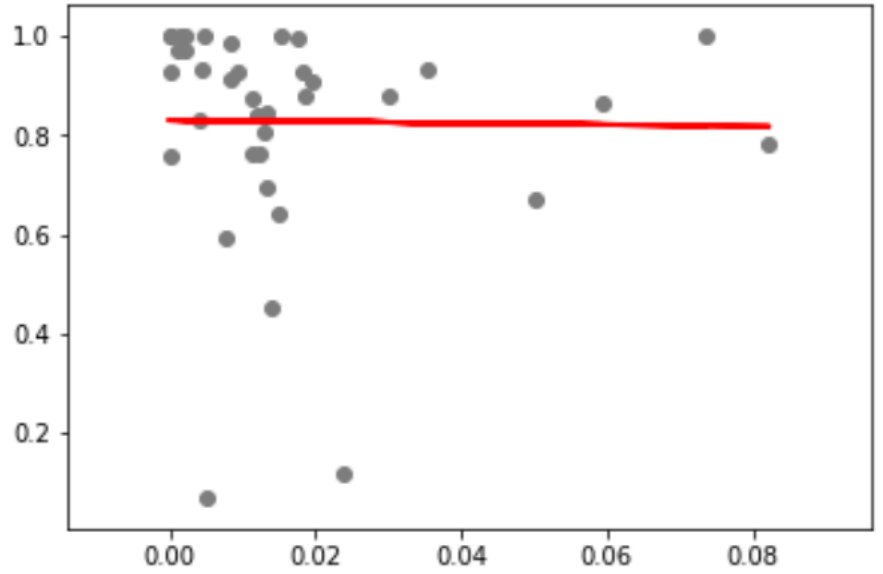
Mean Squared Error: 0.03108102974786959

Tasa de Robos



Mean Squared Error: 0.02442925948158019

Tasa de Asaltos



Mean Squared Error: 0.04716822682927737