



**Universidad Católica del Uruguay**

**Facultad de Ingeniería y Tecnologías**

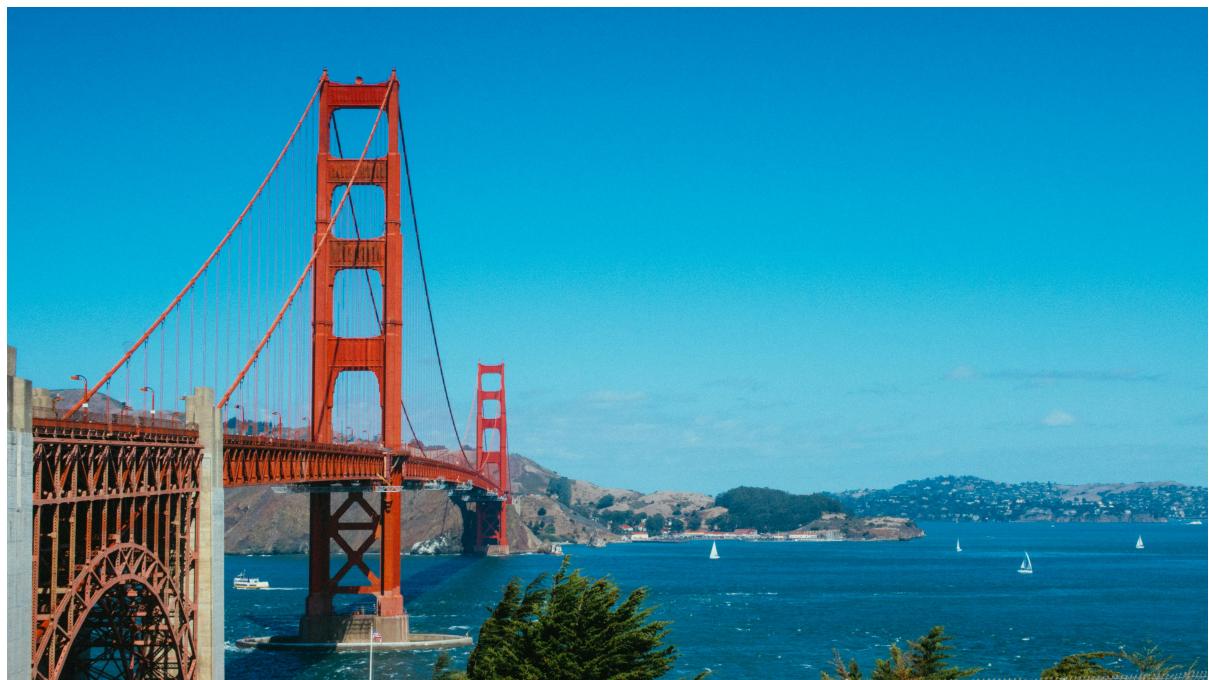
**Inteligencia Artificial 1**

**Alumno:** Juan Manuel Pérez

---

## **Análisis del problema planteado**

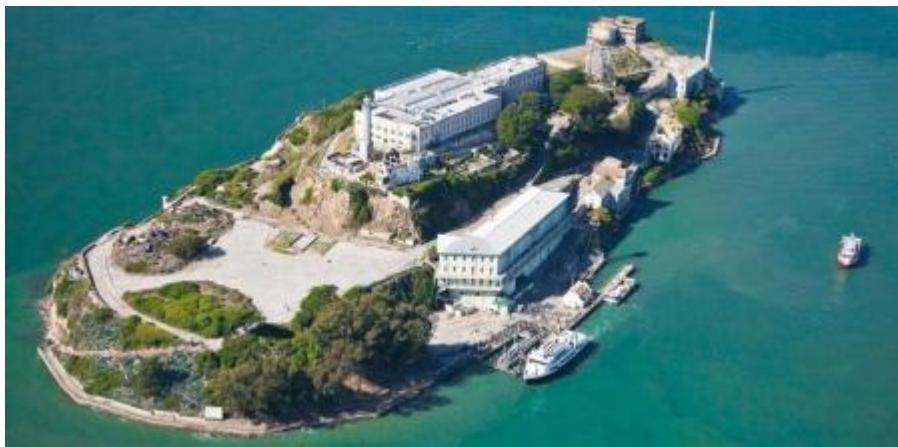
### **Introducción**



El dataset con el cual se hará el presente estudio fue dado por la Cátedra de la Materia Inteligencia Artificial 1 de la Universidad Católica del Uruguay, y se trata del dataset San Francisco Crime Classification, el cual contiene información sobre los crímenes ocurridos en la ciudad de San Francisco, California, Estados Unidos, entre los años 2003 y 2015.

El objetivo de este estudio es el de predecir el tipo de crimen que se cometió, en base a la fecha, hora y locación en la que se cometió el crimen, así como también en base a la descripción del mismo.

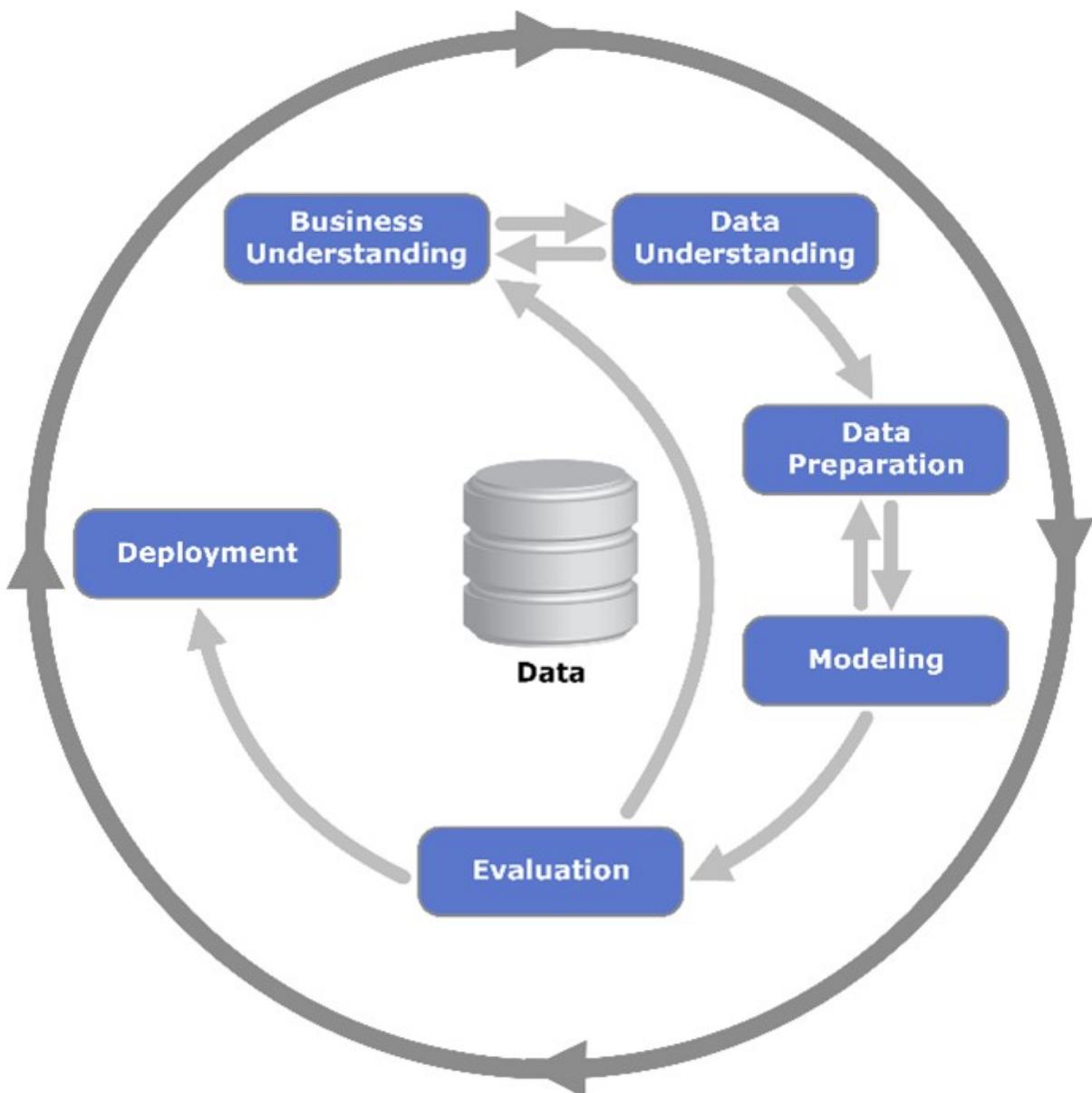
Este es un caso de estudio que surge ya que entre los años 1934 y 1963, San Francisco fue famosa por ser la sede de la prisión federal de Alcatraz, la cual albergaba a los criminales más peligrosos de la época. Entre los presos más famosos se encontraban Al Capone, George "Machine Gun" Kelly y Robert Franklin Stroud, conocido como "El hombre pájaro de Alcatraz".



Aunque podemos decir que hoy en día, la ciudad es reconocida más por su escena tecnológica que por su pasado criminal. Sin embargo, con el aumento de la desigualdad económica, la escasez de viviendas y la proliferación de costosos juguetes digitales que viajan en BART hacia el trabajo, no escasean los actos delictivos en la ciudad junto a la bahía.

## Metodología

Para realizar el presente estudio, se utilizará el lenguaje de programación Python, junto a las librerías de Machine Learning y la herramienta de RapidMiner, para realizar el análisis de los datos y la predicción de los mismos.



Se realizará el estudio acorde al proceso de minería de datos CRISP-DM, el cual se detalla a continuación:

- **Comprendión del negocio:** En esta fase inicial, aquí se busca comprender los objetivos comerciales y las metas del proyecto. Se definen los problemas que se pretende resolver con la minería de datos y se establece un marco general para el proyecto.
- **Comprendión de los datos:** Se recopilan y exploran los datos disponibles para el proyecto. Esto implica la obtención de una visión detallada de la naturaleza de los datos, la identificación de problemas de calidad y la determinación de la idoneidad de los datos para abordar los objetivos comerciales.
- **Preparación de los datos:** Se lleva a cabo el preprocesamiento de los datos. Esto implica la limpieza de datos, la transformación de variables, la selección de atributos relevantes y la creación de conjuntos de datos adecuados para el modelado.
- **Modelado:** En esta etapa es donde se seleccionan y aplican técnicas de modelado para construir y entrenar modelos basados en los datos preparados. Se exploran diferentes algoritmos y se ajustan sus parámetros para encontrar el modelo más efectivo para abordar el problema en cuestión.
- **Evaluación:** Los modelos construidos se evalúan en términos de su eficacia para resolver el problema definido en la fase de Comprendión del Negocio. Se utilizan métricas y técnicas de evaluación para determinar la calidad del modelo y su capacidad para generalizar a nuevos datos.
- **Despliegue:** En la fase final, los modelos evaluados y aprobados se implementan en el entorno operativo. Esto puede implicar la integración con sistemas existentes, la creación de interfaces de usuario o la automatización de procesos basados en los resultados del modelo.

# Objetivo

El objetivo de este estudio es el de predecir el tipo de crimen que se cometió, en base a la fecha, hora y ubicación en la que se cometió el crimen, así como también en base a la descripción del mismo. Para ello contamos con el dataset San Francisco Crime Classification, el cual se encuentra disponible en la plataforma [Kaggle](#), donde se puede apreciar gran cantidad de información referente al mismo.

## Descripción de los datos

Este conjunto de datos contiene incidentes derivados del sistema de informes de incidentes criminales del Departamento de Policía de San Francisco (SFPD, por sus siglas en inglés). Los datos abarcan desde el 1 de enero de 2003 hasta el 13 de mayo de 2015. El conjunto de entrenamiento y el conjunto de prueba rotan cada semana, lo que significa que las semanas 1, 3, 5, 7... pertenecen al conjunto de prueba, mientras que las semanas 2, 4, 6, 8 pertenecen al conjunto de entrenamiento. Esto se hace para tornar los datos resulantes más realistas, ya que en la vida real, los datos no se encuentran ordenados de forma cronológica y separarlos de esta forma nos permite simular mejor la realidad.

## Exploración de datos

En esta sección se realizará el estudio profundo de los datos, para poder comprenderlos mejor y poder realizar un análisis más detallado de los mismos. Como ya se mencionó anteriormente, el dataset esta dividido en dos partes, el conjunto de entrenamiento y el conjunto de prueba.

El conjunto de entrenamiento contiene 878.049 registros y 9 columnas, mientras que el conjunto de prueba contiene 884.262 registros y 6 columnas. Esta diferencia entre las cantidad de columnas se debe a que el conjunto de entrenamiento contiene las columnas "Category", "Descript" y "Resolution", las cuales no se encuentran en el conjunto de prueba, ya que son las columnas que se pretenden utilizar para realizar la predicción.

A continuación se muestra una tabla con las columnas de cada conjunto de datos, así como también una breve descripción de cada una de ellas.

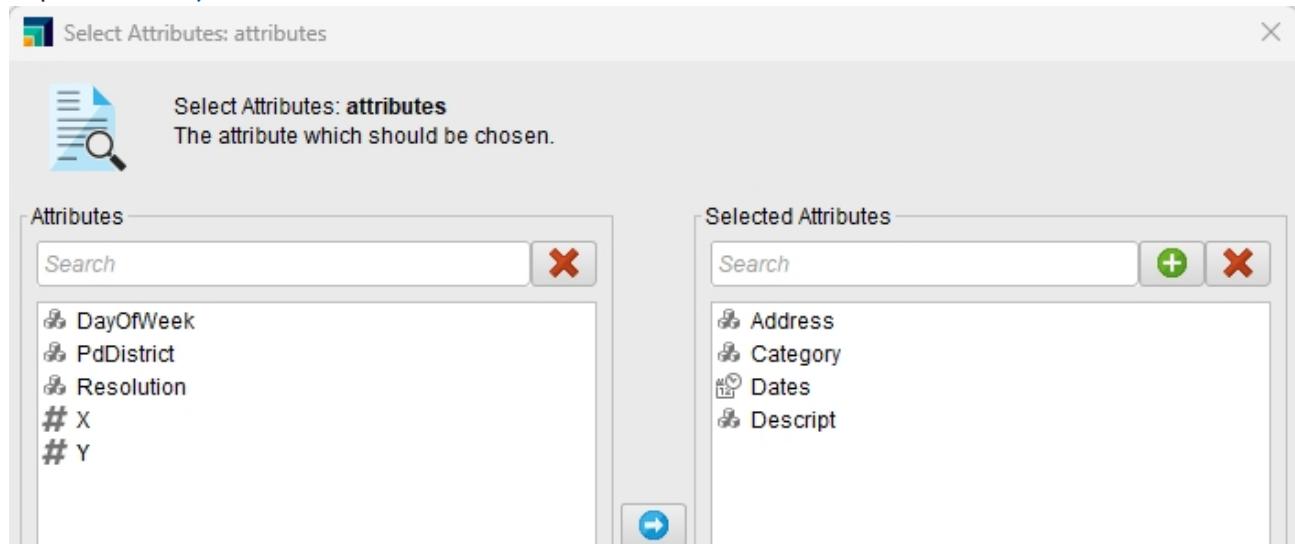
| Columna    | Descripción   | Tipo de dato |
|------------|---|--------------|
| Dates      | Marca de tiempo que indica cuándo se informó el incidente del delito. | DateTime     |
| Category   | Categoría del incidente del crimen. (Solo en Train.csv)               | Polynomial   |
| Descript   | Descripción detallada del incidente del crimen. (Solo en Train.csv)   | Polynomial   |
| DayOfWeek  | El día de la semana en que se informó el incidente del delito.        | Polynomial   |
| PdDistrict | Nombre del distrito del Departamento de Policía                       | Polynomial   |
| Resolution | Cómo se resolvió el incidente del crimen.                             | Polynomial   |
| Address    | La dirección aproximada del incidente del crimen.                     | Polynomial   |
| X          | Coordenada X de la ubicación del incidente del crimen (Longitud).     | Real         |
| Y          | Coordenada Y de la ubicación del incidente del crimen (Latitud).      | Real         |

Los conjuntos de datos no contienen valores nulos o faltantes, por lo que no es necesario realizar un tratamiento de los mismos.

Un aspecto importante que se aprecia en el dataset de entrenamiento es que existen 2692 filas que están duplicadas, esto lo podemos ver porque se aprecian delitos que ocurren en el mismo lugar, a la misma hora y en la misma fecha. Como medida para evitar que estos datos afecten el modelo, se procederá a eliminarlos del dataset, para ello se utilizará la función `drop_duplicates()` de la librería Pandas y el operador Remove Duplicates de RapidMiner.

```
# Eliminación de filas duplicadas. Dates, category, descript y address deben ser iguales para que se considere que son duplicados.
```

```
train.drop_duplicates(subset=['Dates', 'Category', 'Descript', 'Address'],  
inplace=True)
```



En el dataset de prueba no se encontraron filas duplicadas, por lo que no es necesario realizar ningún tratamiento sobre los mismos.

Cuando se observaron los datos de las coordenadas, se pudo apreciar que existen valores que no corresponden a la ciudad de San Francisco, por lo que se procedió a eliminarlos del dataset, para ello se utilizó la función `drop()` de la librería Pandas y el operador Filter Examples de RapidMiner.

```
# Eliminación de filas que no corresponden a San Francisco. Se eliminan los valores que se encuentran fuera de los siguientes rangos:
```

```
# X: menores o iguales a -121.5, que comprenden las Y: menores o iguales a 37.820
```

```
train.drop(train[(train['X'] <= -121.5) | (train['Y'] <= 37.820)].index,  
inplace=True)
```

En el caso del dataset de entrenamiento, se eliminaron 67 filas y en el caso del dataset de prueba, se eliminaron 78 filas. Hasta el momento los dataset cuentan con 875290 y 884184 registros respectivamente.

El siguiente paso es colocar el indicador de la variable objetivo en el dataset de prueba, para ello se utilizó el operador Set Role de RapidMiner, para asignarle el rol de "label" a la columna "Category", de esta forma, el modelo sabrá que esta es la variable que se pretende predecir.

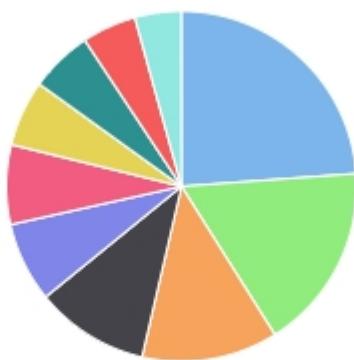
Al mismo tiempo aprovechamos para eliminar las columnas que no se utilizarán para el estudio, en este caso se eliminaron las columnas "Descript" y "Resolution", ya que no aportan información relevancia al modelo que se pretende realizar, para ello se utilizó el operador Select Attributes de RapidMiner.

Como resultado de realizar estas operaciones ahora ambos datasets (train y test) cuentan con 7 columnas.

Por último, para poder realizar el estudio de los datos, se observó la clase objetivo (Category), para ver cuantos valores distintos existen y cuantos registros hay de cada uno de ellos. El atributo Category cuenta con 39 valores distintos, de los cuales se listan a continuación los 10 más frecuentes.

| No. | Categoría      | Cantidad de registros |
|-----|----------------|-----------------------|
| 1   | LARCENY/THEFT  | 174263                |
| 2   | OTHER OFFENSES | 125913                |
| 3   | NON-CRIMINAL   | 91889                 |
| 4   | ASSAULT        | 76787                 |
| 5   | DRUG/NARCOTIC  | 53902                 |
| 6   | VEHICLE THEFT  | 53664                 |
| 7   | VANDALISM      | 44566                 |
| 8   | WARRANTS       | 42133                 |
| 9   | BURGLARY       | 36594                 |
| 10  | SUSPICIOUS OCC | 31386                 |

### Top 10 crímenes



- LARCENY/THEFT
- NON-CRIMINAL
- DRUG/NARCOTIC
- VANDALISM
- BURGLARY

- OTHER OFFENSES
- ASSAULT
- VEHICLE THEFT
- WARRANTS
- SUSPICIOUS OCC

Tomando en consideración la información que acabamos de plasmar, podemos ver que nos encontramos ante un problema de clasificación multiclas, ya que la variable objetivo cuenta con 39 valores distintos. Conocer esta información es importante, ya que nos permite saber que tipo de modelo debemos utilizar para realizar la predicción, teniendo en cuenta que se trata de un problema de

clasificación multiclas. Los 2 modelos que se utilizarán para realizar la predicción son: Random Forest y Naive Bayes.

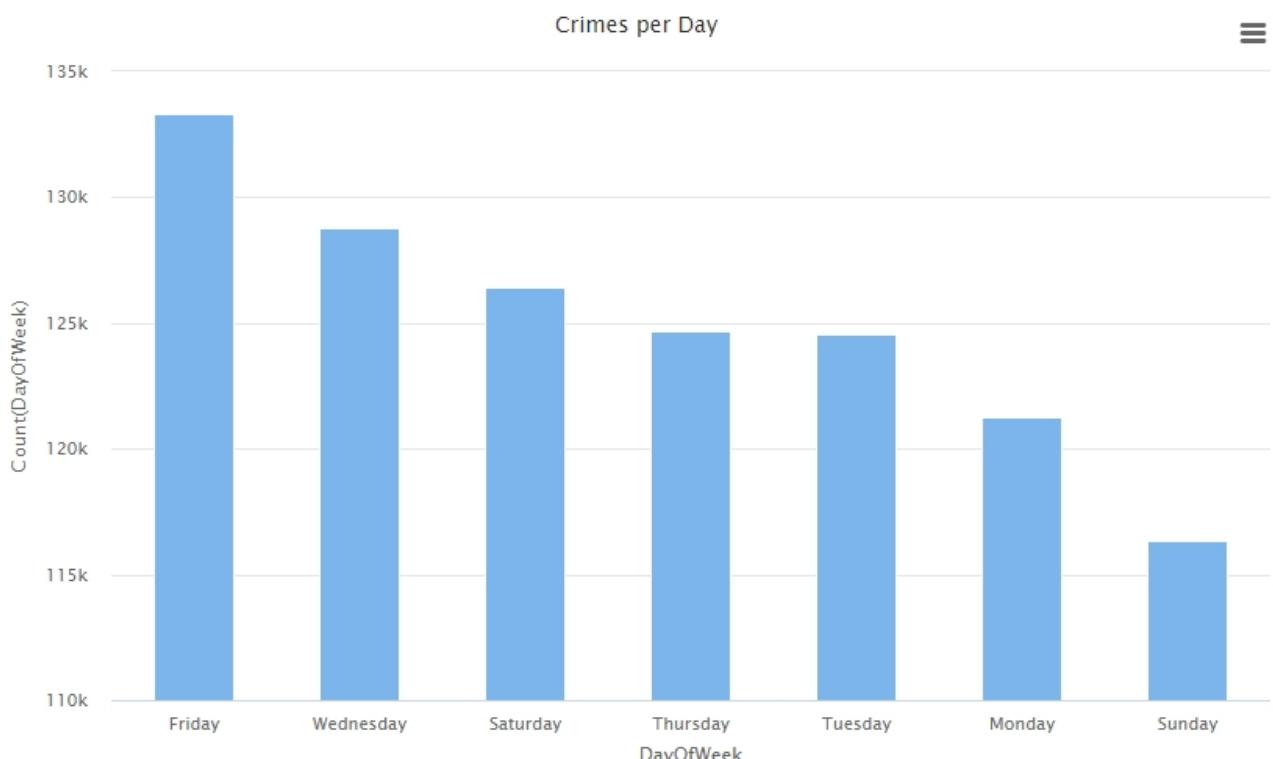
Se escoge Random Forest porque es un modelo que se utiliza para problemas de clasificación y regresión, y se escoge Naive Bayes porque es un modelo que se utiliza para problemas de clasificación multiclas. En ambos casos serán desarrollados tanto en Python como en RapidMiner, para poder comparar los resultados obtenidos en cada uno de ellos.

## Datos interesantes

Al realizar diferentes mediciones de los datos, se encuentran resultados como la frecuencia de crímenes según el día de la semana, la hora con mayores picos de crímenes, los distritos con mayor cantidad de crímenes, entre otros. A continuación se muestran algunos de estos datos.

### Día de la semana con mayor cantidad de crímenes

| Día de la semana | Cantidad de crímenes |
|------------------|----------------------|
| Viernes          | 133299               |
| Miércoles        | 128767               |
| Sábado           | 126410               |
| Jueves           | 124705               |
| Martes           | 124543               |
| Lunes            | 121230               |
| Domingo          | 116336               |

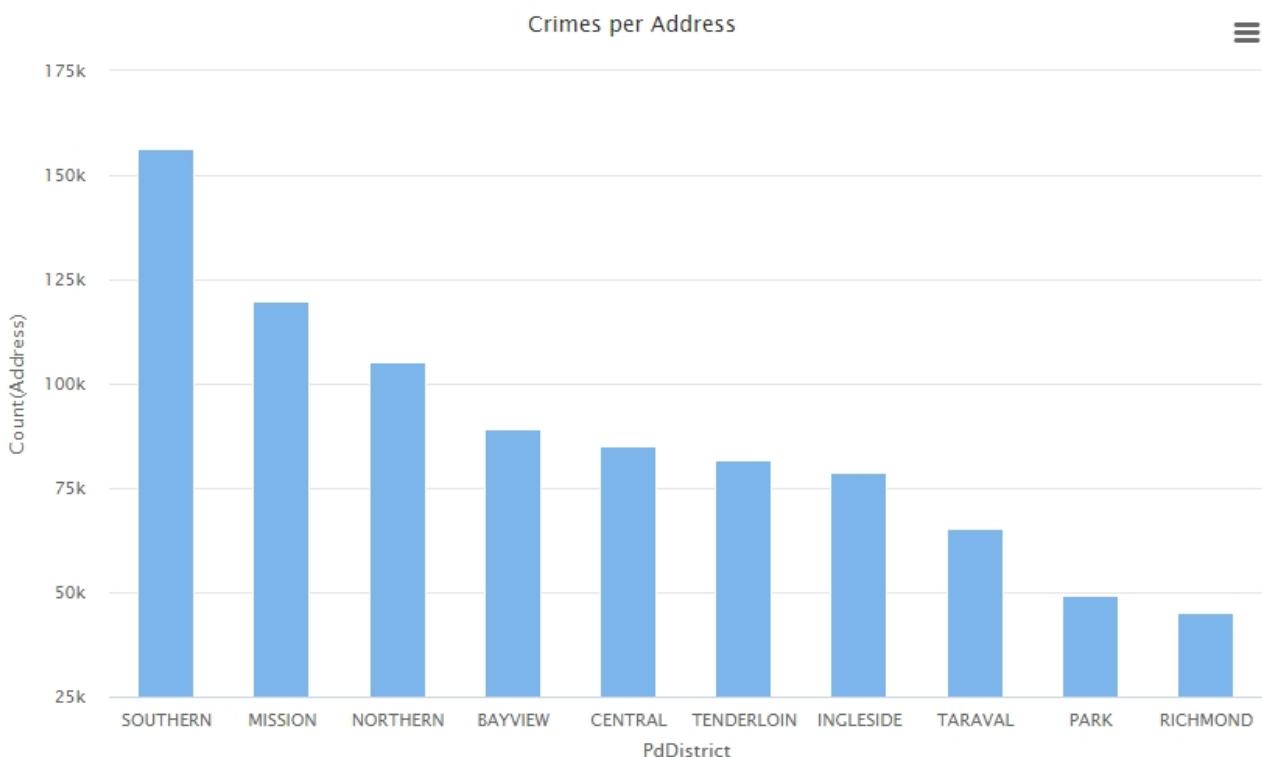


Se pueden apreciar que los días con mayor cantidad de crímenes son los viernes y los miércoles, mientras que el día con menor cantidad de crímenes es el domingo. Esto se puede deber a que los viernes y miércoles son los días en los que la gente sale más a la calle, ya sea para ir a trabajar o para salir a divertirse, mientras que los domingos es un día en el que la gente suele quedarse en sus casas.

## Cantidad de crímenes por distrito

Dentro de los datos útiles para poder comprender mejor el dataset, se encuentra la cantidad de crímenes por distrito, ya que nos permite saber en que distritos se cometen más crímenes y en cuales se cometan menos. A continuación se muestra una tabla con los 10 distritos y su cantidad de crímenes respectivos.

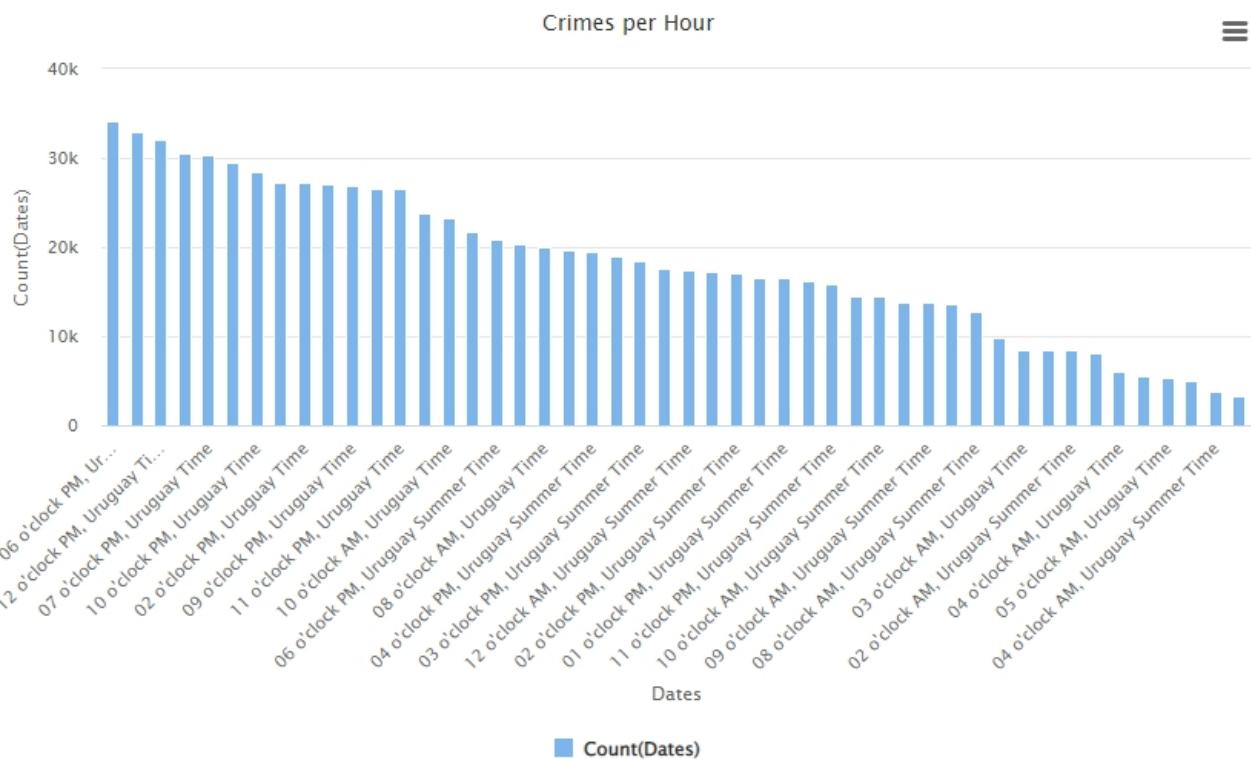
| Distrito   | Cantidad de crímenes |
|------------|----------------------|
| SOUTHERN   | 156446               |
| MISSION    | 119685               |
| NORTHERN   | 105065               |
| BAYVIEW    | 89016                |
| CENTRAL    | 85252                |
| TENDERLOIN | 81604                |
| INGLESIDE  | 78694                |
| TARAVAL    | 65351                |
| PARK       | 49127                |
| RICHMOND   | 45060                |



Como puede apreciarse en la imagen, el distrito con mayor cantidad de crímenes es el distrito SOUTHERN, mientras que el distrito con menor cantidad de crímenes es el distrito RICHMOND. Si bien no es asunto de este trabajo, podría investigarse las causas de estos resultados, ya que podría ser de utilidad para las autoridades de la ciudad de San Francisco, para poder tomar medidas que ayuden a reducir la cantidad de crímenes en los distritos con mayor cantidad de crímenes.

**Otros 2 datos interesantes son Crímenes por hora y Crímenes por fecha.**

Los crímenes por hora nos permiten saber a que hora del día se cometen más crímenes, mientras que los crímenes por fecha nos permiten saber en que fecha se cometieron más crímenes. A continuación se muestran los resultados de estudiar estos datos.



Nominal values

| Index | Nominal value | Absolute count | Fraction |
|-------|---------------|----------------|----------|
| 1     | 01/11         | 5314           | 0.006    |
| 2     | 25/01         | 5253           | 0.006    |
| 3     | 22/02         | 4991           | 0.006    |
| 4     | 11/01         | 4955           | 0.006    |
| 5     | 08/02         | 4941           | 0.006    |
| 6     | 04/04         | 4863           | 0.006    |
| 7     | 08/03         | 4840           | 0.006    |
| 8     | 19/04         | 4834           | 0.006    |
| 9     | 31/10         | 4829           | 0.006    |
| 10    | 10/01         | 4812           | 0.005    |

Datos a valorar de estos 2 casos, el día que más crímenes se cometieron fueron los 1 de noviembre, seguidos por los 25 de enero y en tercer lugar los 22 de febrero. En cuanto a la hora, la hora en la que más crímenes se cometieron fueron a las 18:00 horas y a las 12:00 horas, mientras que la hora en la que menos crímenes se cometieron fue a las 04:00 horas.

## Preparación de los datos

En esta sección se realizará el preprocesamiento de los datos, para poder prepararlos para el modelado. Para ello se utilizará la librería Pandas de Python y la herramienta RapidMiner.

Por motivos visuales se utilizará la herramienta RapidMiner para realizar el preprocesamiento de los datos, ya que permite visualizar los datos de una forma más amigable, sin embargo, se mostrará el código de Python utilizado para realizar el preprocesamiento de los datos.

## Transformación de variables

En esta sección se realizará la transformación de las variables, para ello se utilizará la herramienta RapidMiner y la librería Pandas de Python.

### Transformación de la variable "Dates"

En rapidminer se utilizó el operador Generate Attributes para crear 4 nuevas columnas, las cuales son: "Year", "Month", "Day" y "Hour". Estas columnas se crearon a partir de la columna "Dates", la cual contiene la fecha y hora en la que se cometió el crimen. El separar estas columnas nos permite poder realizar un análisis más detallado de los datos, ya que nos permite saber en que año, mes, día y hora se cometió el crimen, permitiendo así poder realizar un análisis más detallado de los datos.

| column name | function expressions                                    |
|-------------|---|
| Year        | date_get(Dates, DATE_UNIT_YEAR, "America/Los_Angeles")  |
| Month       | date_get(Dates, DATE_UNIT_MONTH, "America/Los_Angeles") |
| Day         | date_get(Dates, DATE_UNIT_DAY, "America/Los_Angeles")   |
| Hour        | date_get(Dates, DATE_UNIT_HOUR, "America/Los_Angeles")  |

En el caso de Python, se utilizó la función `to_datetime()` de la librería Pandas, para convertir la columna "Dates" en una columna de tipo DateTime, y luego se utilizó la función `dt.year`, `dt.month`, `dt.day` y `dt.hour` para crear las columnas "Year", "Month", "Day" y "Hour" respectivamente.

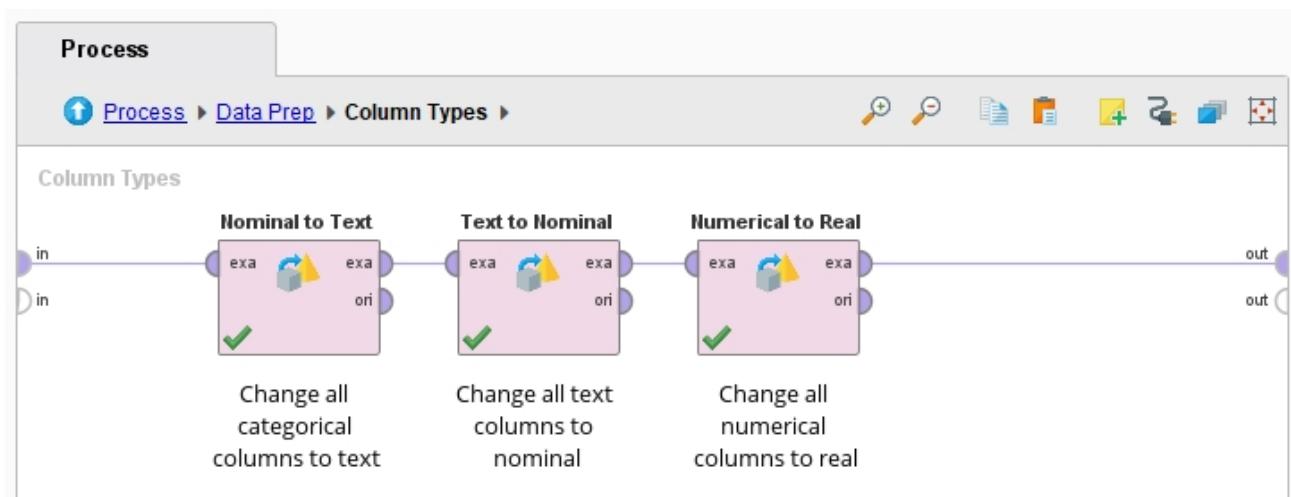
```
# Transformación de la columna "Dates" a tipo DateTime
train['Dates'] = pd.to_datetime(train['Dates'])

# Creación de las columnas "Year", "Month", "Day" y "Hour"
train['Year'] = train['Dates'].dt.year
train['Month'] = train['Dates'].dt.month
train['Day'] = train['Dates'].dt.day
train['Hour'] = train['Dates'].dt.hour
```

## Transformación de tipos de datos

Las columnas poseen diferentes tipos de datos por eso se procede a transformarlas a los tipos de datos que corresponden. Para ello se utilizaron los operadores de RapidMiner Nominal to Text, Text to Nominal

y Numerical to Real, para transformar las columnas a los tipos de datos que corresponden, todas estas transformaciones se encuentran en el subprocesso "Column Types" en Data Preparation.



En el caso de Python, se utilizó la función astype() de la librería Pandas, para transformar las columnas a los tipos de datos que corresponden.

```
# Transformación de Los tipos de datos
train['Category'] = train['Category'].astype('category')
train['DayOfWeek'] = train['DayOfWeek'].astype('category')
train['PdDistrict'] = train['PdDistrict'].astype('category')
train['Address'] = train['Address'].astype('category')
train['Year'] = train['Year'].astype('int64')
train['Month'] = train['Month'].astype('int64')
train['Day'] = train['Day'].astype('int64')
train['Hour'] = train['Hour'].astype('int64')
# X e Y son valores reales, por lo que se transforman a tipo float64
train['X'] = train['X'].astype('float64')
train['Y'] = train['Y'].astype('float64')
```

## Remover duplicados

En esta sección se procede a remover los duplicados del dataset, para ello se utilizó el operador Remove Duplicates de RapidMiner, con el cual se eliminaron las filas que coincidían en las columnas "Dates", "Category", "Descript" y "Address".

Realizar esta operación es importante, ya que nos permite eliminar los datos que no aportan información relevante al modelo, ya que son datos que se repiten y no aportan información nueva. Sin dudas esto ayuda a mejorar el modelo, ya que al eliminar estos datos, se reduce el tiempo de procesamiento innecesario.

## Selección de atributos

En esta sección se procede a seleccionar los atributos que se utilizarán para realizar el modelado, para ello se utilizó el operador Select Attributes de RapidMiner, con el cual se seleccionaron las columnas que no se utilizarán para realizar el modelado, las cuales son: "Dates", "Descript" y "Resolution".

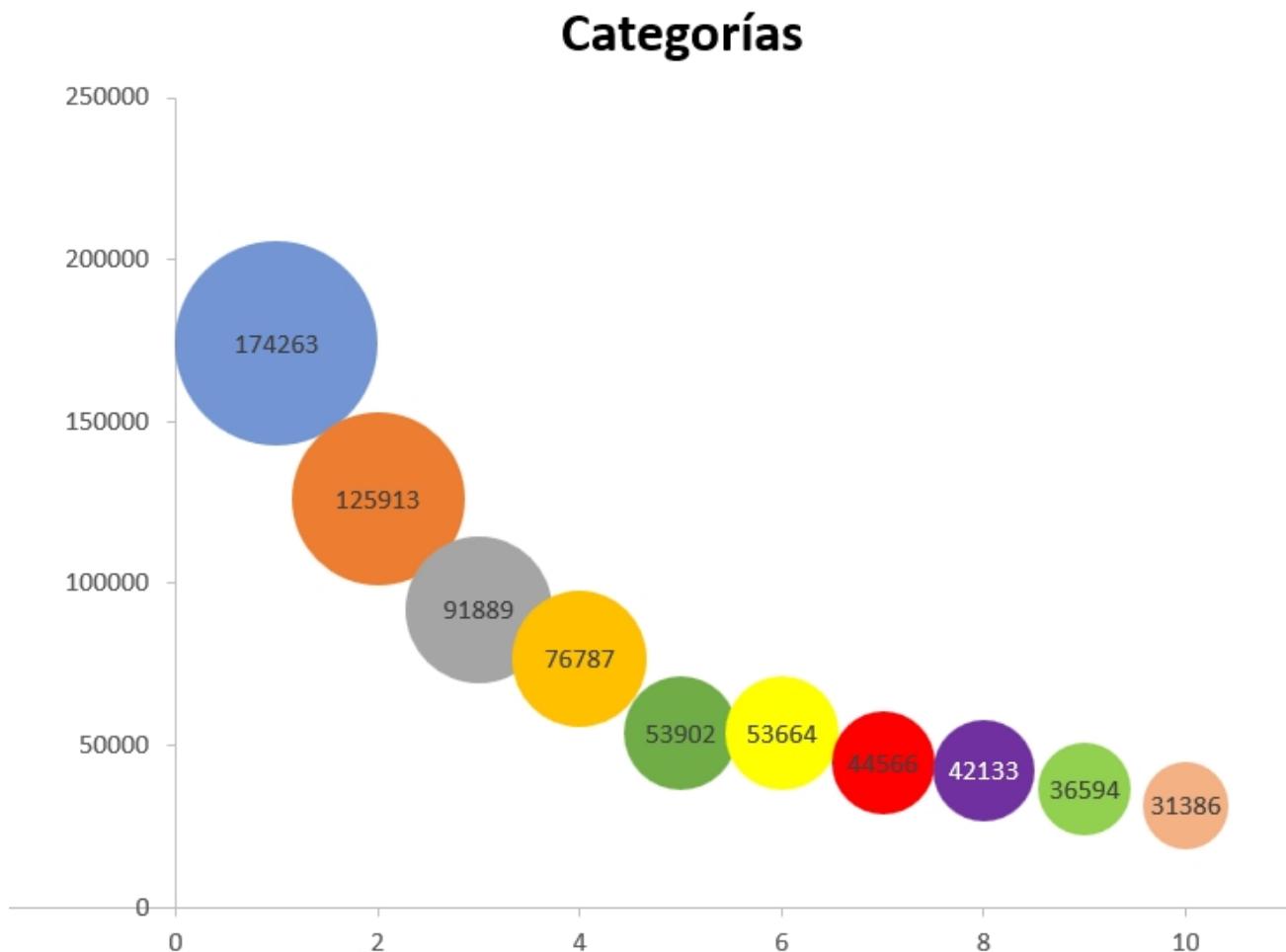
Descript y Resolution no se utilizarán para realizar el modelado, ya que son columnas que no aportan información relevante al modelo, ya que son descripciones del crimen y como se resolvió el mismo, mientras que la columna "Dates" se eliminó porque se crearon 4 columnas nuevas a partir de ella, las cuales son: "Year", "Month", "Day" y "Hour", las cuales se utilizarán para realizar el modelado.

# Selección de la variable objetivo

En esta sección se procede a seleccionar la variable objetivo, para ello se utilizó el operador Set Role de RapidMiner, con el cual se le asignó el rol de "label" a la columna "Category", la cual es la variable que se pretende predecir.

Como se mencionó anteriormente, la variable objetivo es la columna "Category", ya que es la variable que se pretende predecir, esta columna cuenta con 39 valores distintos, los cuales son los distintos tipos de crímenes que se cometieron en la ciudad de San Francisco entre los años 2003 y 2015.

## Top 10 Categorías de crímenes



# Modelado

En esta sección se procede a realizar el modelado de los datos, para ello se utilizarán los modelos Random Forest y Naive Bayes, los cuales se desarrollarán tanto en Python como en RapidMiner, para poder comparar los resultados obtenidos en cada uno de ellos.

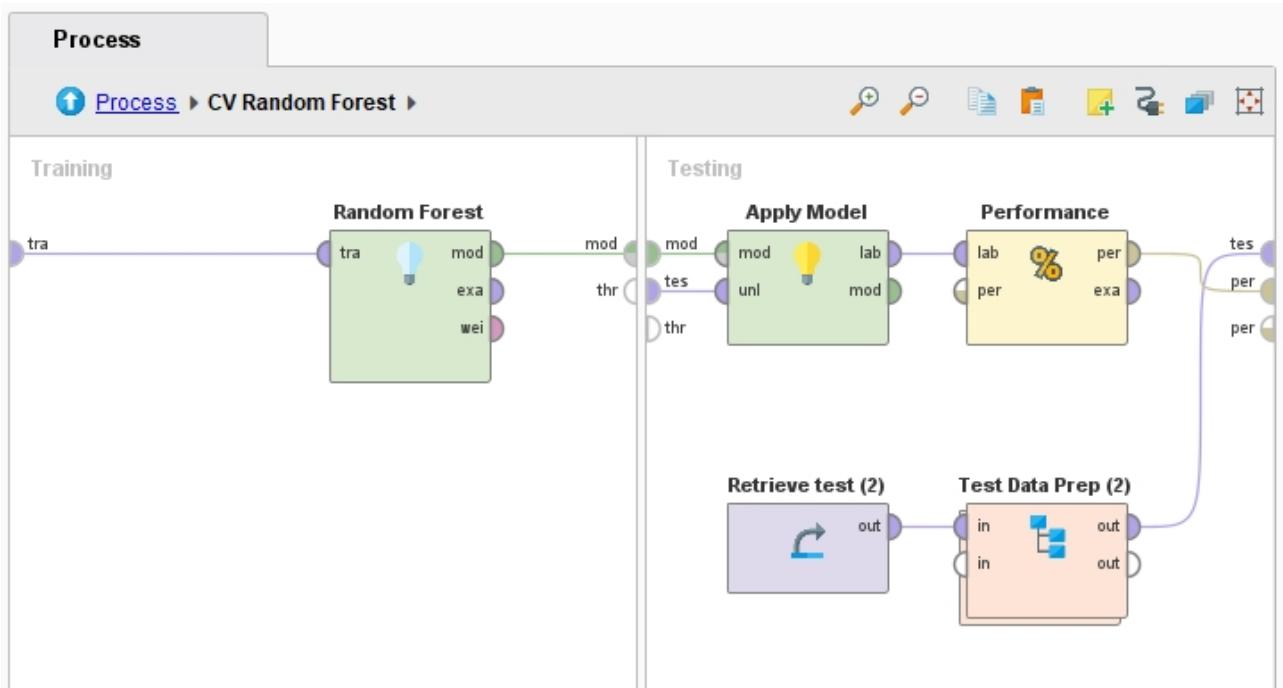
## Random Forest

Random Forest es un algoritmo de aprendizaje supervisado. El "bosque" que crea Random Forest está formado por un gran número de árboles de decisión individuales que operan como un conjunto. Cada árbol individual en el bosque genera una predicción de clase y la clase con más votos se convierte en la predicción de nuestro modelo.

Es un algoritmo versátil que se puede utilizar tanto para problemas de clasificación como de regresión y que proporciona muy buenos resultados incluso sin ajustar los parámetros del modelo. Además, Random Forest es un algoritmo muy robusto y preciso debido a la cantidad de árboles que se utilizan en el proceso de predicción.

## RapidMiner Random Forest (Prueba reducida)

En la etapa de comprobación de pruebas de los algoritmos se utilizaron los operadores de RapidMiner: Split Data, Random Forest, Apply Model y Performance (Classification). Para la validación se utilizó el método de validación cruzada con 9 folds, el cual consiste en dividir el conjunto de datos en 9 partes iguales, de las cuales 8 se utilizan para entrenar el modelo y 1 para probarlo, este proceso se repite 9 veces, de forma que cada parte se utiliza para probar el modelo una vez.



Los parámetros utilizados para el Cross Validation fueron los siguientes:

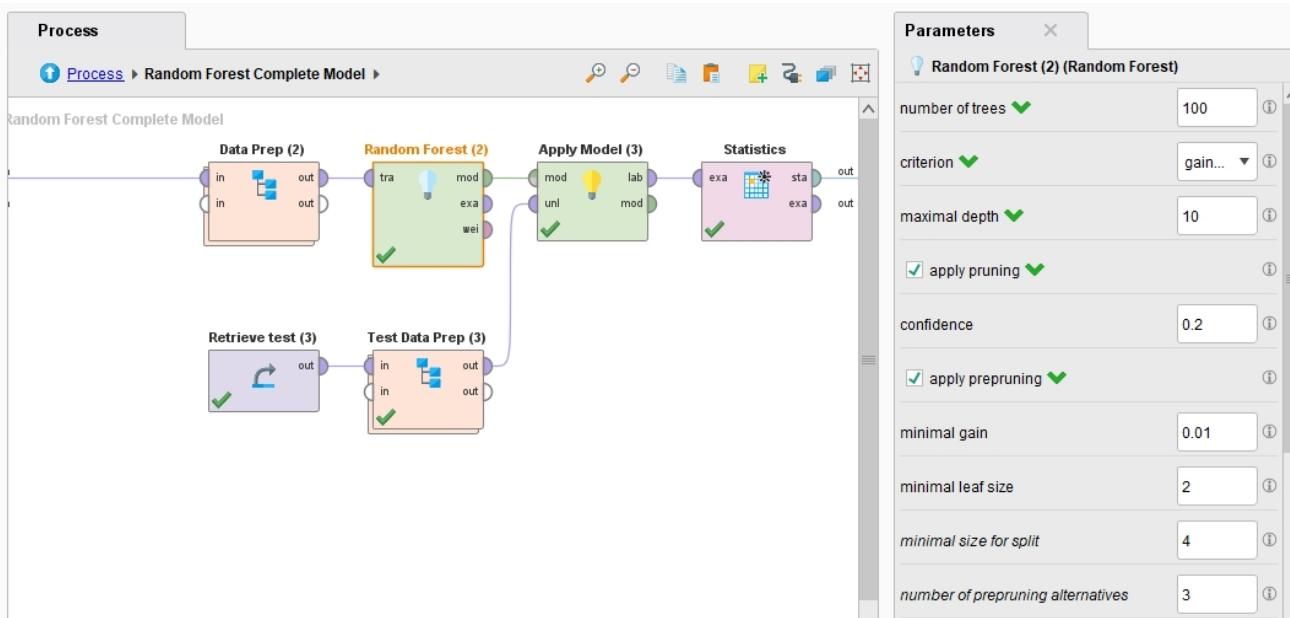
- **Number of Folds:** 9
- **Seed:** 1992
- **Sampling Type:** Stratified Sampling
- **Enable parallel execution:** True

En el propio modelo de Random Forest se utilizaron los siguientes parámetros:

- **Number of Trees:** 100
- **Criterion:** Accuracy
- **Maximal Depth:** 10
- **Apply Pruning:** True *Confidence:* 0.1
- **Apply Pre-Pruning:** True *Minimum Gain:* 0.01 *Minimum Leaf Size:* 2 *Minimum Size for Split:* 4 *Number of Prepruning Alternatives:* 3

Con esta configuración se logra un valor de **Accuracy** de **23.42% +/- 0.39%**, lo cual si bien no es un valor muy alto, es un valor aceptable, teniendo en cuenta que se trata de un problema de clasificación multiclase, en el cual se pretende predecir 39 valores distintos.

## RapidMiner Random Forest (Prueba completa)



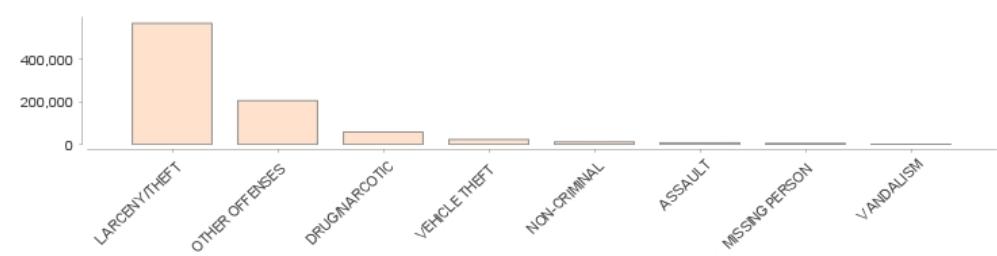
Al hacer una ejecución con todos los datos del dataset, se obtuvo un valor de **Estabilidad** de predicción del 64.62% con una **Validez** del 35.38%, con datos que indican que los valores predichos se encuentran en 29 de las 39 categorías existentes.

### < > prediction(Category)

#### Summary

Category  
 Missing: 0.00%  
 Infinite: 0.00%  
 ID-ness: 0.00%  
 Stability: 64.62%  
 Valid: 35.38%

#### Top Values



#### 29 Distinct Values:

| Value          | Count   | Percentage |
|----------------|---------|------------|
| LARCENY/THEFT  | 569,027 | 64.36%     |
| OTHER OFFENSES | 205,814 | 23.28%     |
| DRUG/NARCOTIC  | 58,277  | 6.59%      |
| VEHICLE THEFT  | 23,440  | 2.65%      |
| NON-CRIMINAL   | 11,664  | 1.32%      |
| ASSAULT        | 8,036   | 0.91%      |

Como se puede apreciar en la imagen superior, la predicción indica que 569.027 casos son para la categoría LARCENY/THEFT. Esto se debe a que es la categoría que más se repite en el dataset, por lo que el modelo tiende a predecir esta categoría, ya que es la que más se repite en el dataset.

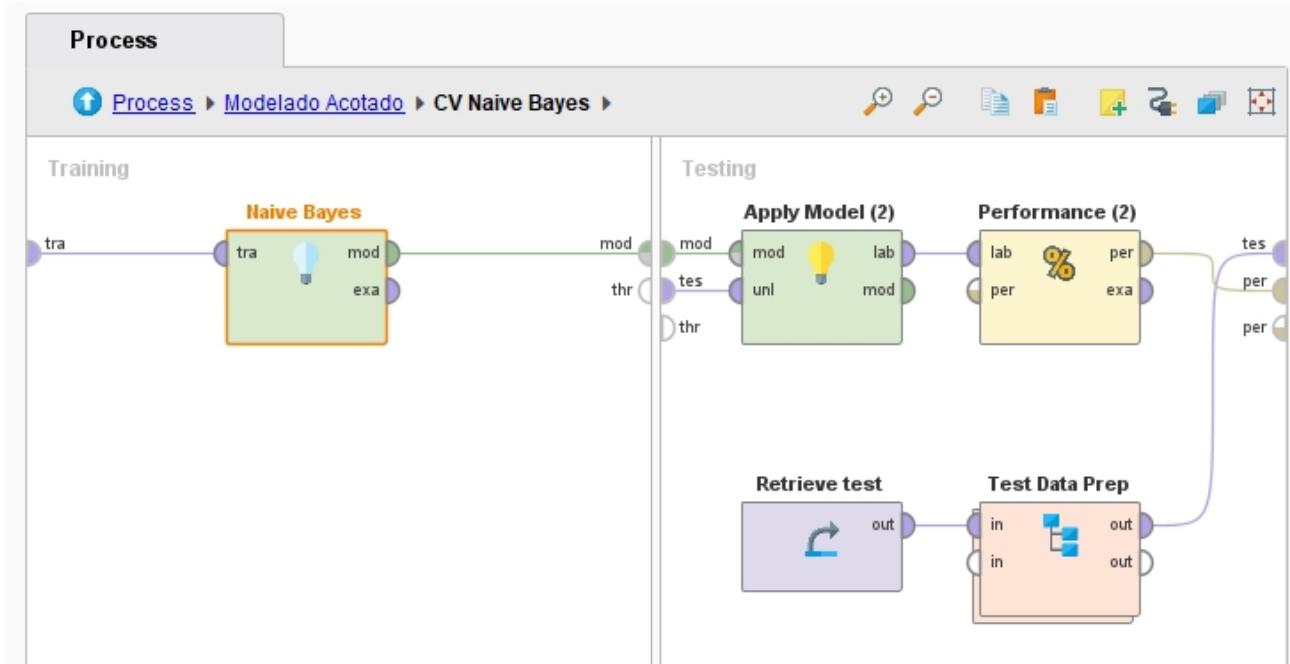
## Naive Bayes

Naive Bayes es un algoritmo de aprendizaje supervisado basado en el teorema de Bayes. Es un algoritmo simple pero poderoso que se utiliza ampliamente para la clasificación de texto (debido a su alta precisión) y con menos frecuencia en la regresión y la predicción. El clasificador Naive Bayes asume que el efecto de una característica particular en una clase determinada es independiente de otras características. Por ejemplo, un fruto puede considerarse como una manzana si es rojo, redondo y de alrededor de 3 pulgadas de diámetro. Incluso si estas características dependen entre sí o de la existencia

de las otras características, todas estas propiedades contribuyen de forma independiente a la probabilidad de que este fruto sea una manzana y es por eso que se llama algoritmo Naive Bayes.

## RapidMiner Naive Bayes (Prueba reducida)

En la etapa de comprobación de pruebas de los algoritmos se utilizaron los operadores de RapidMiner: Split Data, Naive Bayes, Apply Model y Performance (Classification). Para la validación se utilizó el método de validación cruzada con 9 folds, el cual consiste en dividir el conjunto de datos en 9 partes iguales, de las cuales 8 se utilizan para entrenar el modelo y 1 para probarlo, este proceso se repite 9 veces, de forma que cada parte se utiliza para probar el modelo una vez.



Los parámetros utilizados para el Cross Validation fueron los siguientes:

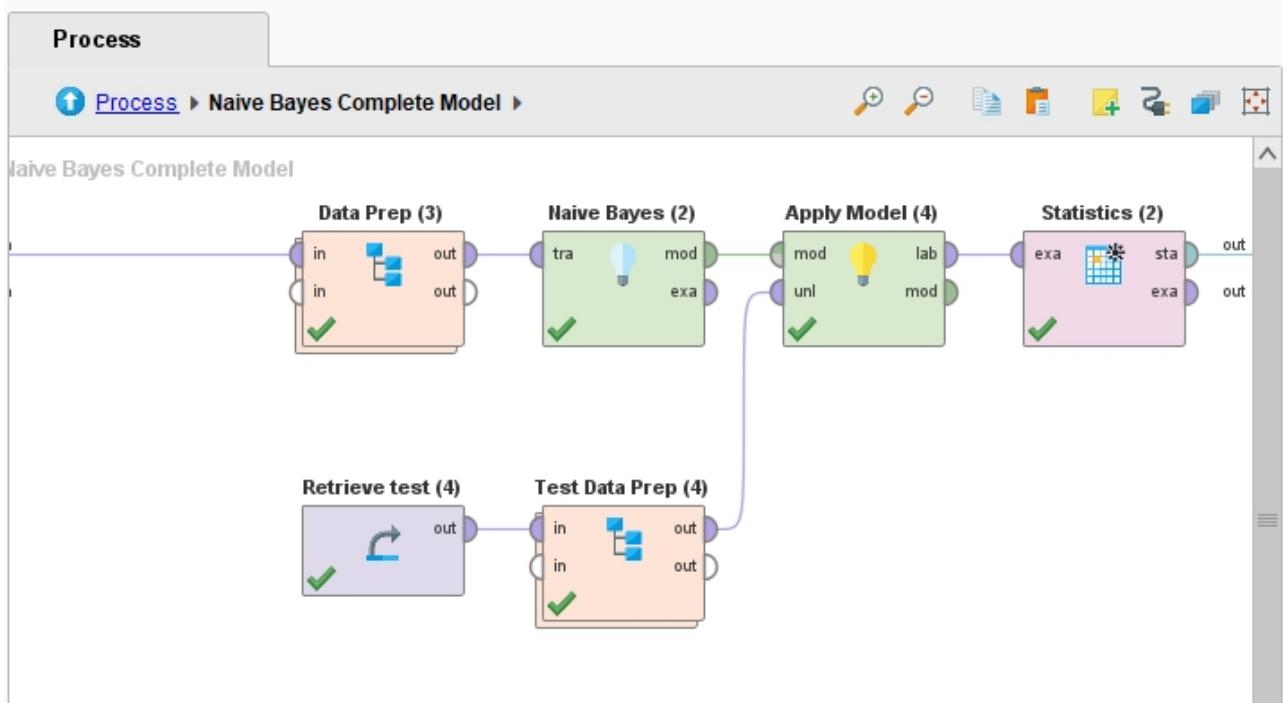
- **Number of Folds:** 9
- **Seed:** 1992
- **Sampling Type:** Stratified Sampling
- **Enable parallel execution:** True

En el propio modelo de Naive Bayes se utilizaron los siguientes parámetros:

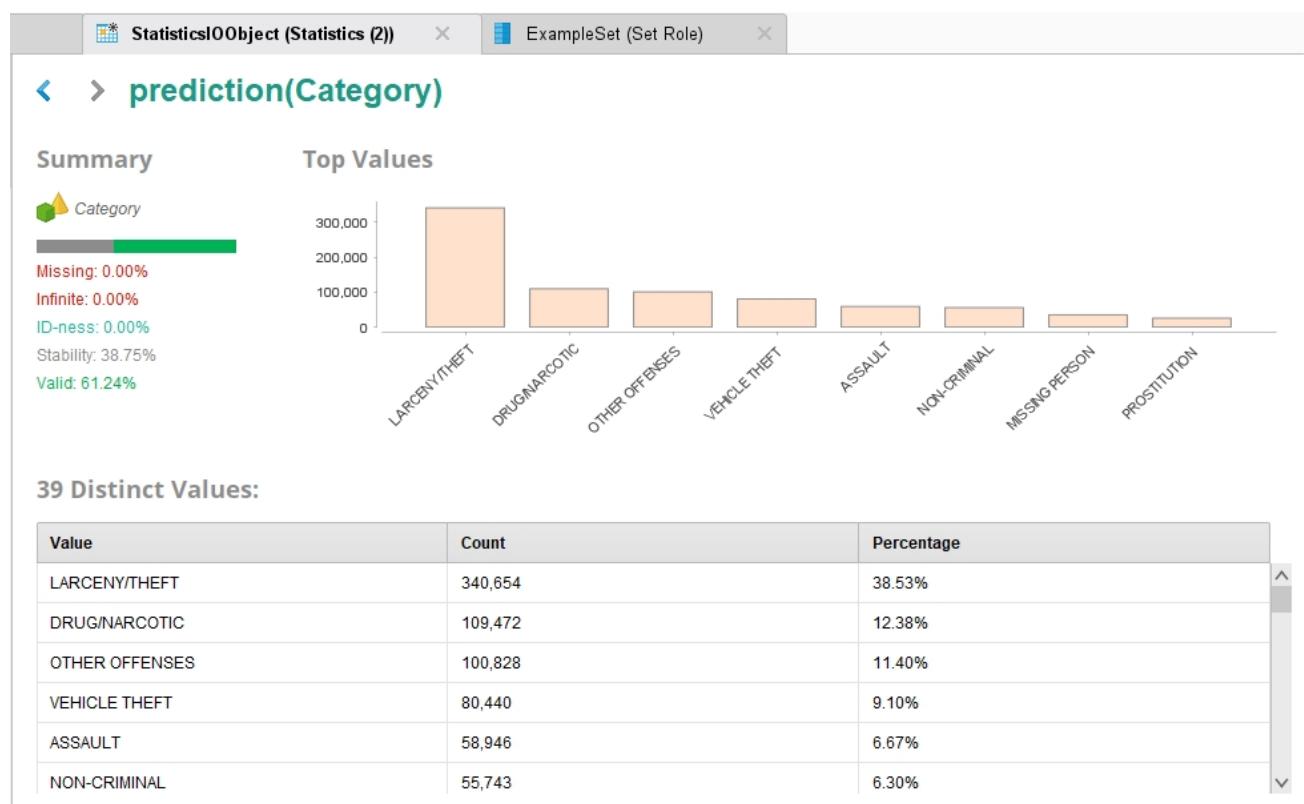
- **Laplace Correction:** True

El utilizar esta configuración se logra un valor de **Accuracy** de **20.16% +/- 0.47%**, lo cual si bien no es un valor muy alto, es un valor aceptable, teniendo en cuenta que se trata de un problema de clasificación multiclase, en el cual se pretende predecir 39 valores distintos.

## RapidMiner Naive Bayes (Prueba completa)



Al hacer una ejecución con todos los datos del dataset, se obtuvo un valor de **Estabilidad** de predicción del 38.75% con una **Validez** del 61.24%, con datos que indican que los valores predichos se encuentran en las 39 categorías existentes.



Lo que se puede apreciar en la imagen superior, la predicción indica que 340.654 casos son para la categoría LARCENY/THEFT. Esto se debe a que es la categoría que más se repite en el dataset, por lo que el modelo tiende a predecir esta categoría, ya que es la que más se repite en el dataset.

## Evaluación

En esta sección se pretende evaluar los modelos desarrollados, para ello se utilizarán las métricas de evaluación Accuracy y Confusion Matrix, las cuales nos permitirán evaluar los modelos y comparar los resultados obtenidos en cada uno de ellos.

# Accuracy

La métrica de evaluación Accuracy nos permite evaluar el modelo, ya que nos permite saber que tan preciso es el modelo, es decir, que tan bien predice el modelo. Esta métrica se calcula dividiendo la cantidad de predicciones correctas entre la cantidad total de predicciones.

La métrica de evaluación Accuracy se calcula de la siguiente forma:

- **Accuracy:**  $(TP + TN) / (TP + TN + FP + FN)$

Donde:

- **TP:** True Positive
- **TN:** True Negative
- **FP:** False Positive
- **FN:** False Negative

En los Modelos Random Forest y Naive Bayes se obtuvieron los siguientes valores de Accuracy:

| Modelo        | Accuracy Rapid Miner | Accuracy Python |
|---------------|----------------------|-----------------|
| Random Forest | 23.42%               | 27.53%          |
| Naive Bayes   | 20.16%               | 20.74%          |

Como se puede apreciar en la tabla anterior, el modelo que obtuvo un mejor valor de Accuracy fue el modelo Random Forest, con un valor de 23.42%, mientras que el modelo Naive Bayes obtuvo un valor de 20.16%.

# Confusion Matrix

La matriz de confusión es una herramienta que permite la visualización del desempeño de un algoritmo que se emplea en aprendizaje supervisado. Cada fila de la matriz representa las instancias en una clase predicha, mientras que cada columna representa las instancias en una clase real. El nombre proviene del hecho de que permite ver si el sistema está confundiendo dos clases (es decir, si las está etiquetando incorrectamente).

La matriz de confusión se representa de la siguiente forma:

- **True Positive:** Predicción correcta de la clase positiva.
- **True Negative:** Predicción correcta de la clase negativa.
- **False Positive:** Predicción incorrecta de la clase positiva.
- **False Negative:** Predicción incorrecta de la clase negativa.

La tabla de confusión se representa de la siguiente forma:

|                | Predictión Positiva | Predictión Negativa |
|----------------|---------------------|---------------------|
| Clase Positiva | True Positive       | False Negative      |
| Clase Negativa | False Positive      | True Negative       |

Los modelos presentaron los siguientes resultados:

```
# Se muestran los top 10 resultados del Random Forest por cantidad de crímenes.  
# los datos se sacan de model_rf_pred_total  
  
print("Top 10 resultados del Random Forest por cantidad de crímenes")  
print(pd.Series(model_rf_pred_total).value_counts().head(10))  
[26] ✓ 0.0s  
.. Top 10 resultados del Random Forest por cantidad de crímenes  
LARCENY/THEFT      546554  
OTHER OFFENSES     160512  
DRUG/NARCOTIC      67542  
ASSAULT             44185  
VEHICLE THEFT       29537  
NON-CRIMINAL        16071  
MISSING PERSON      10108  
PROSTITUTION        8864  
RUNAWAY              303  
TRESPASS              225  
Name: count, dtype: int64
```

```
# De igual manera, se muestran los top 10 resultados del Naive Bayes por cantidad de crímenes.  
  
print("Top 10 resultados del Naive Bayes por cantidad de crímenes")  
print(pd.Series(model_nb_pred_total).value_counts().head(10))  
[27] ✓ 0.1s  
.. Top 10 resultados del Naive Bayes por cantidad de crímenes  
LARCENY/THEFT      692052  
OTHER OFFENSES     135111  
ASSAULT             43338  
VEHICLE THEFT       10968  
NON-CRIMINAL        2419  
BURGLARY             298  
GAMBLING              76  
Name: count, dtype: int64
```

## Random Forest

Como se aprecia en imagen superior el modelo de Random Forest realizado en Python, obtuvo diferentes valores a los obtenidos en RapidMiner, esto se produce por la configuración propia de cada herramienta. Ya que los valores estaban ajustados de la misma manera, pero al ser herramientas distintas, los resultados obtenidos son distintos.

## Naive Bayes

De igual modo pasa con el modelo de Naive Bayes, ya que los valores obtenidos en Python son distintos a los obtenidos en RapidMiner, esto se debe a la configuración propia de cada herramienta.

*Nota: la única diferencia presente es que hubo que ajustar distintos los datasets en python ya que no aceptan variables categoricas como en los días de la semana, los distritos y las direcciones.*

## Conclusiones

En este trabajo se realizó un estudio sobre el dataset San Francisco Crime Classification, el cual contiene información sobre los crímenes ocurridos en la ciudad de San Francisco, California, Estados Unidos, entre los años 2003 y 2015.

El objetivo de este estudio es el de predecir el tipo de crimen que se cometió, en base a la fecha, hora y ubicación en la que se cometió el crimen, así como también en base a la descripción del mismo.

Para ello se utilizaron los modelos Random Forest y Naive Bayes, los cuales se desarrollaron tanto en Python como en RapidMiner, para poder comparar los resultados obtenidos en cada uno de ellos.

El ejercicio realizado en Python se encuentra en el siguiente en [Parcial2Python](#) y los archivos utilizados para rapidminer se encuentran en la carpeta [RapidMiner](#).