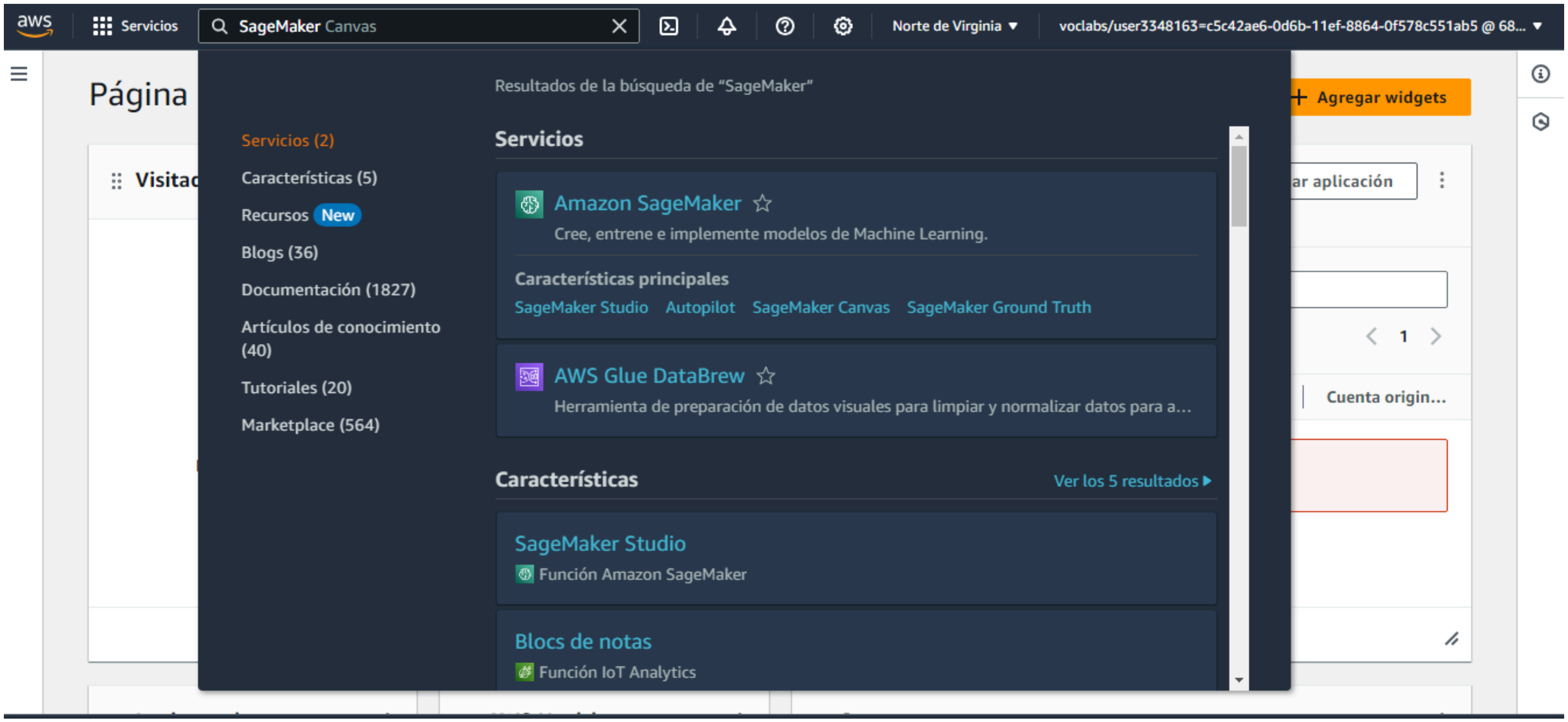
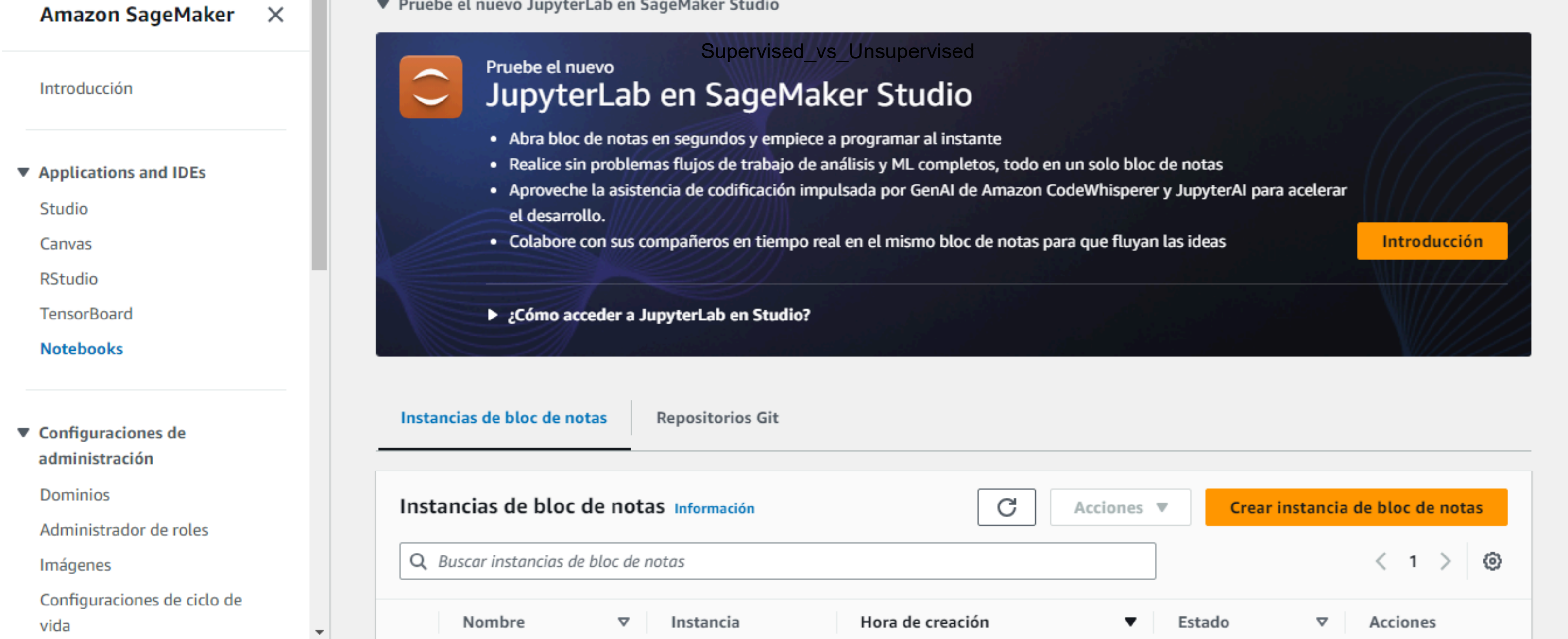
# Exercise: Supervised vs Unsupervised

- In this exercise, you will complete two machine learning tasks:

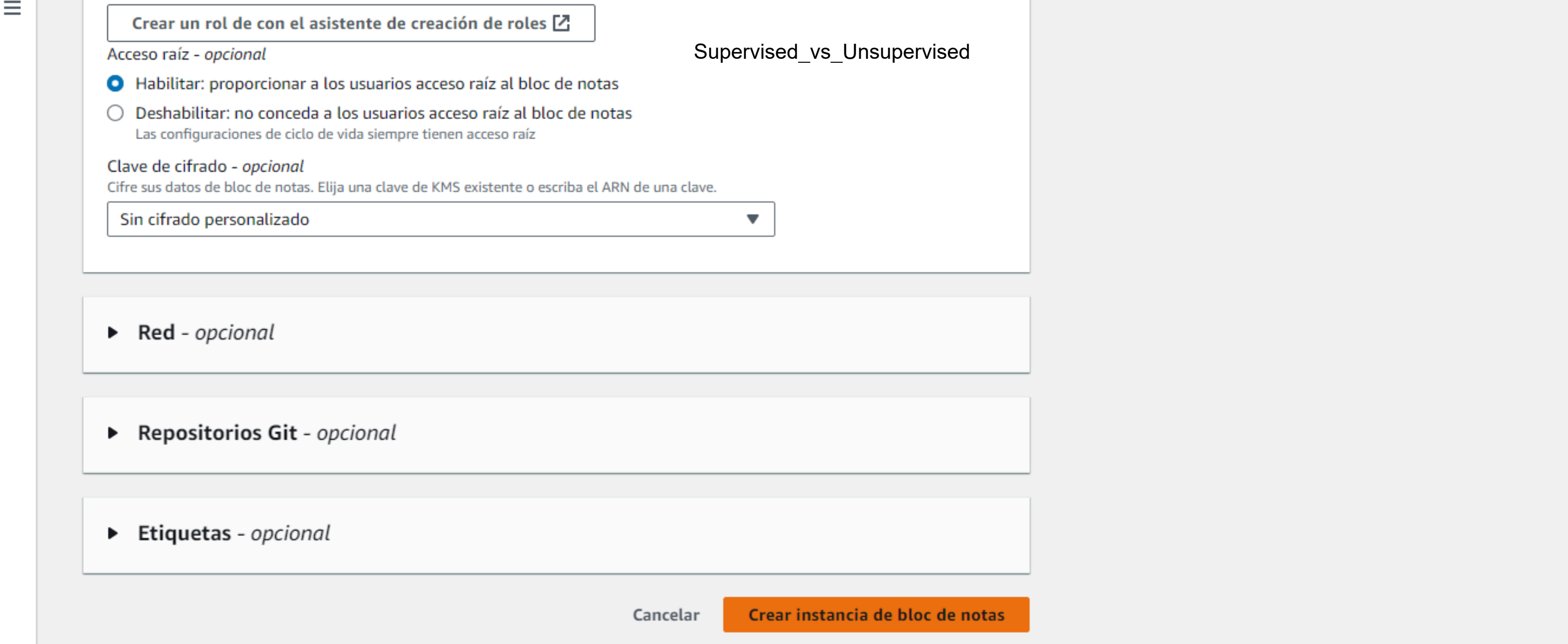
1. Click Launch **Cloud Gateway** in the bottom of the navigation menu
2. From the AWS console, search for **SageMaker**



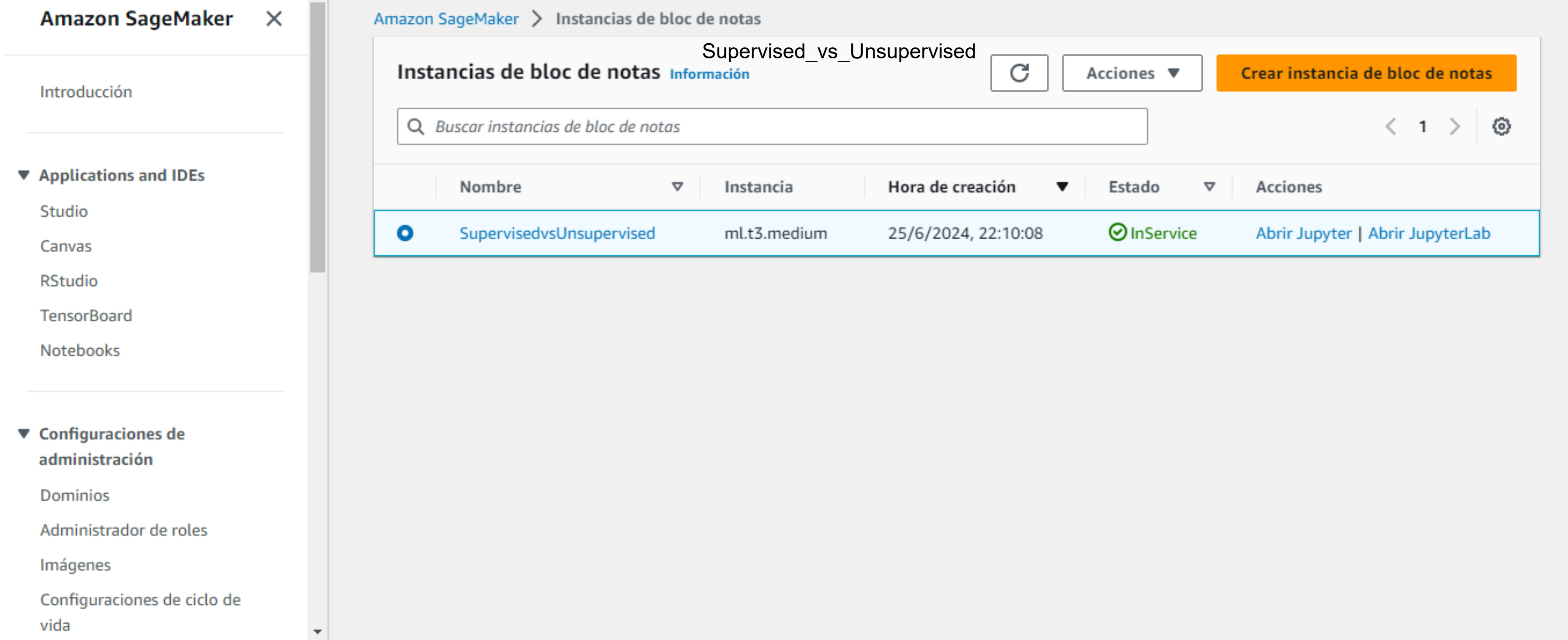

- In the left side menu click **Notebooks**, then click **Notebook instances**



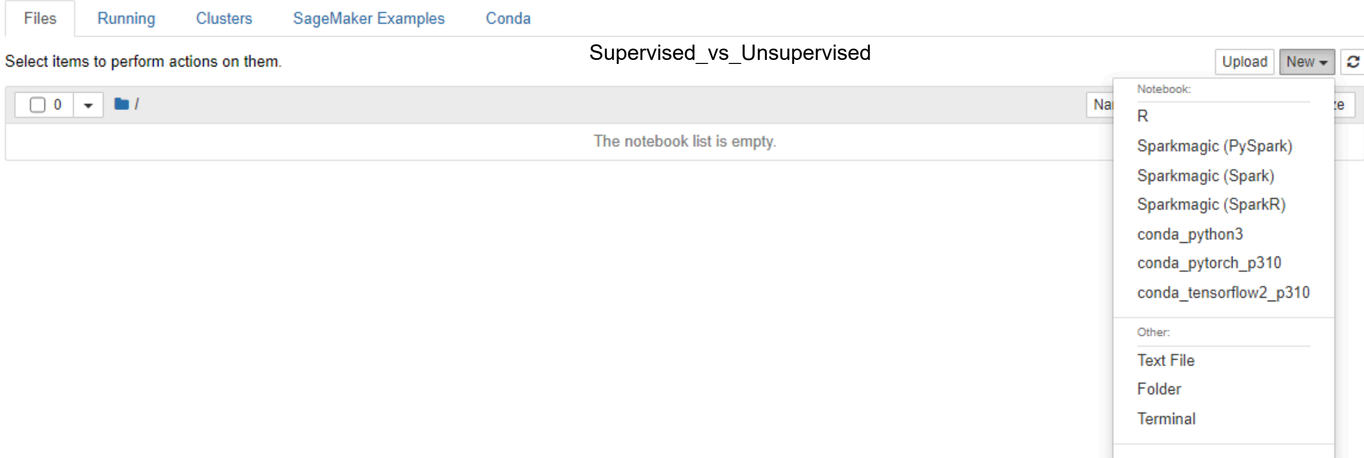

3. Click **Create notebook instance**




4. When the instance is ready, click **Open Jupyter**

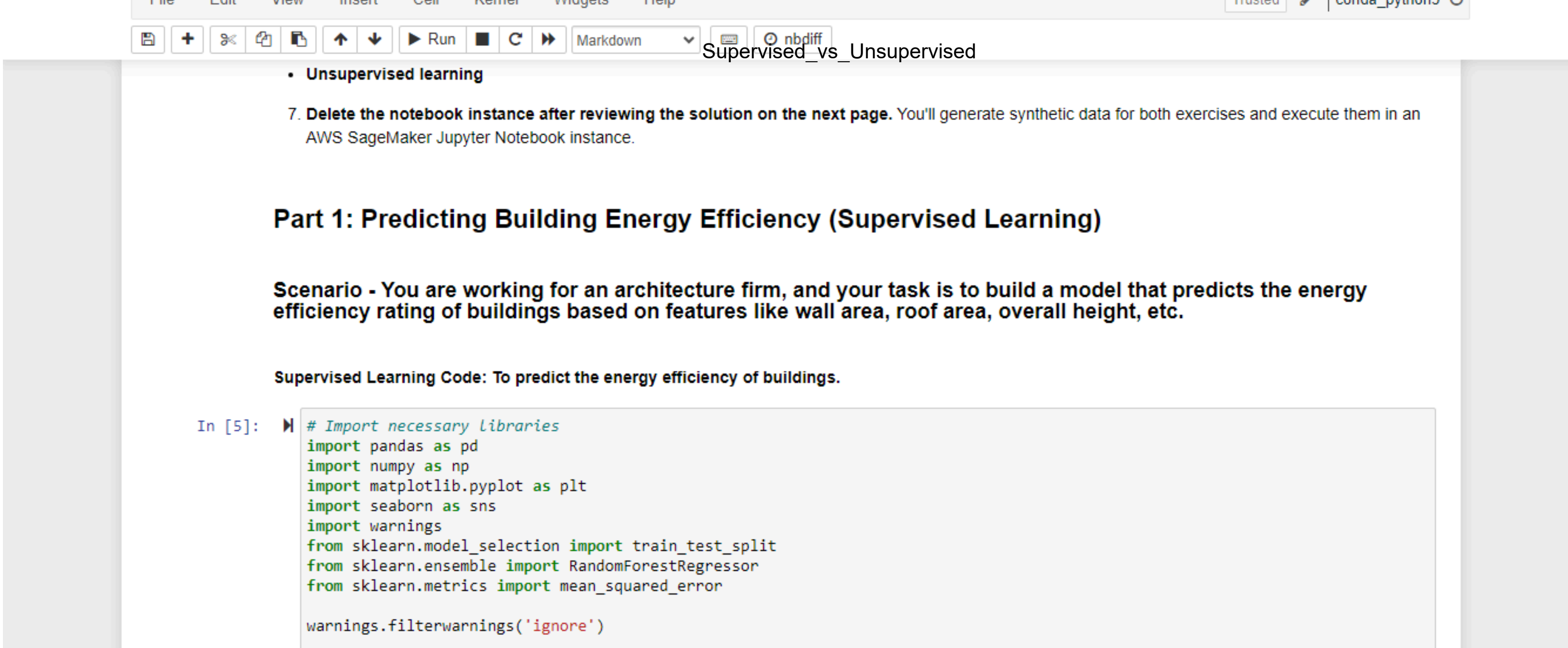



5. When in the notebook, click **New** and select **conda_python3**




6. Copy the code below on this page and paste it into a notebook cell, then run the cell

- **Supervised learning**
- **Unsupervised learning**



- **Unsupervised learning**

7. **Delete the notebook instance after reviewing the solution on the next page.** You'll generate synthetic data for both exercises and execute them in an AWS SageMaker Jupyter Notebook instance.

**Part 1: Predicting Building Energy Efficiency (Supervised Learning)**

**Scenario - You are working for an architecture firm, and your task is to build a model that predicts the energy efficiency rating of buildings based on features like wall area, roof area, overall height, etc.**

**Supervised Learning Code: To predict the energy efficiency of buildings.**

In [5]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error

warnings.filterwarnings('ignore')
```

# Part 1: Predicting Building Energy Efficiency (Supervised Learning)

## Scenario - You are working for an architecture firm, and your task is to build a model that predicts the energy efficiency rating of buildings based on features like wall area, roof area, overall height, etc.

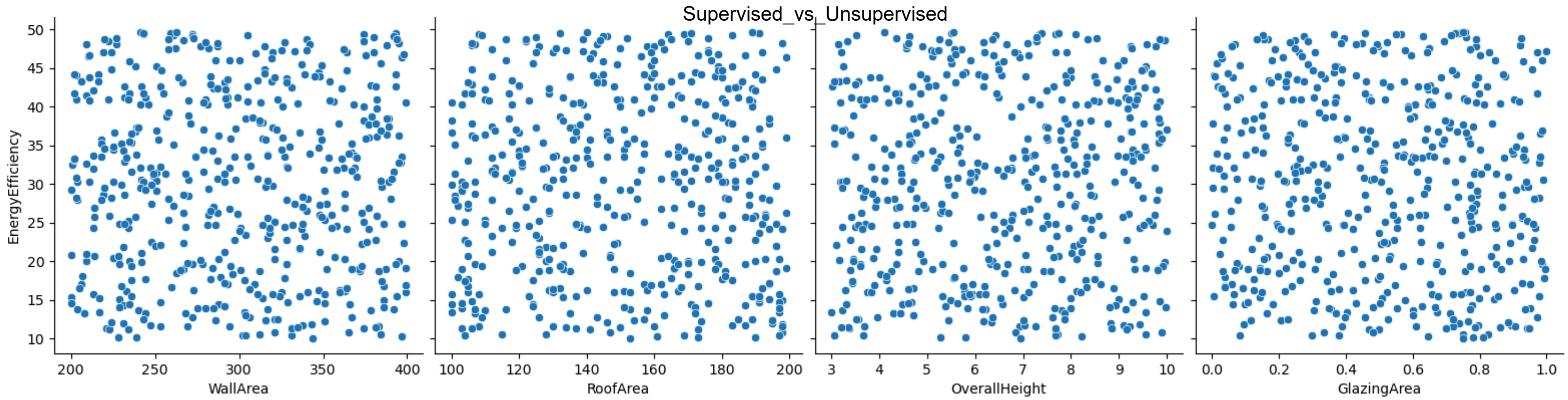
### Supervised Learning Code: To predict the energy efficiency of buildings.

In [ ]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from sklearn.model_selection import train_test_split
```

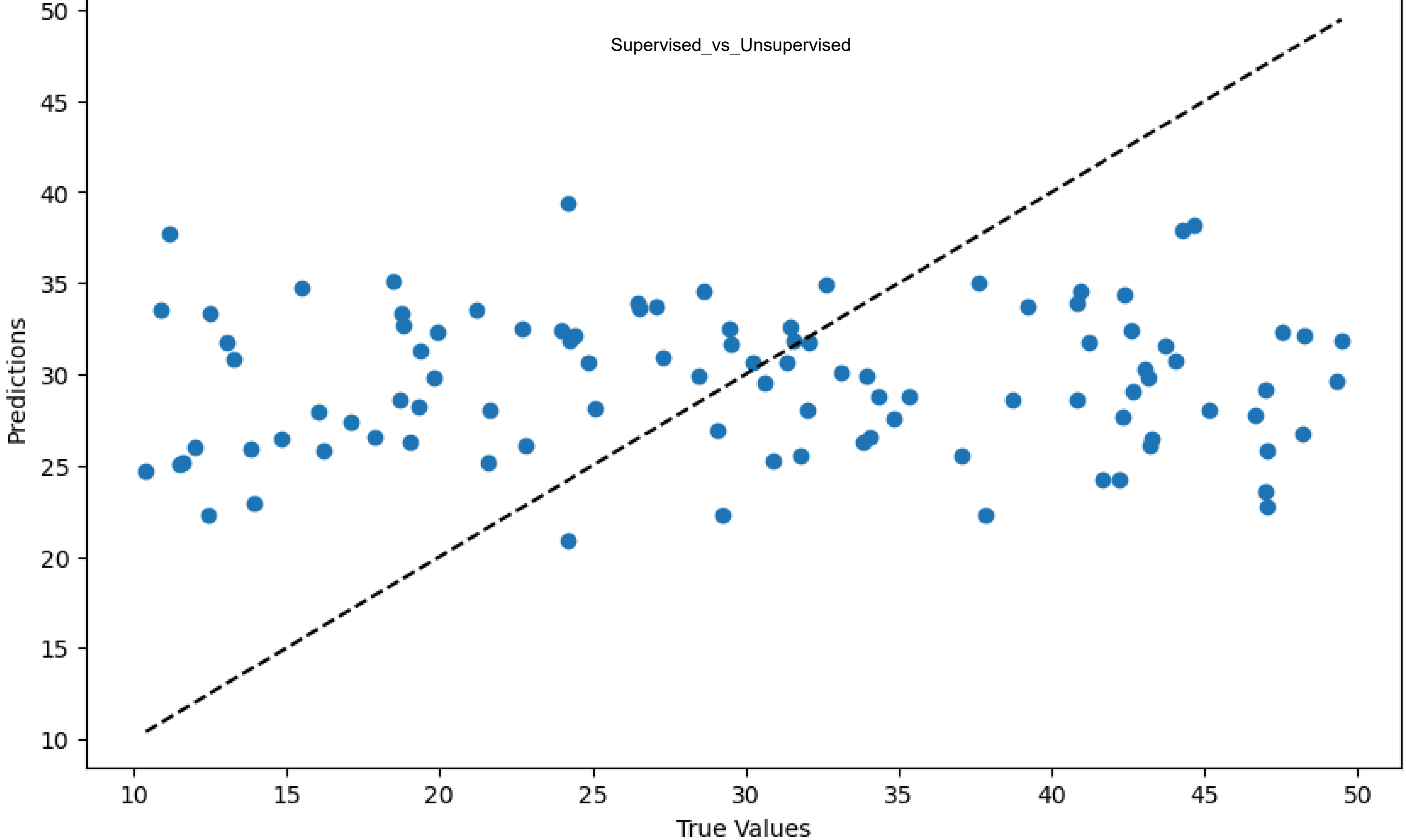
```
warnings.filterwarnings('ignore')

# Generate synthetic dataset for building features and energy efficiency ratings
np.random.seed(0)
data_size = 500
data = {
    'WallArea': np.random.randint(200, 400, data_size),
    'RoofArea': np.random.randint(100, 200, data_size),
    'OverallHeight': np.random.uniform(3, 10, data_size),
    'GlazingArea': np.random.uniform(0, 1, data_size),
    'EnergyEfficiency': np.random.uniform(10, 50, data_size)  # Energy efficiency rating
}
df = pd.DataFrame(data)

# Data preprocessing
X = df.drop('EnergyEfficiency', axis=1)
y = df['EnergyEfficiency']

# Visualize the relationships between features and the target variable (Energy Efficiency)
sns.pairplot(df, x_vars=['WallArea', 'RoofArea', 'OverallHeight', 'GlazingArea'], y_vars='EnergyEfficiency', height=4
plt.show()

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a Random Forest model
model = RandomForestRegressor()
model.fit(X_train, y_train)

# Predict and evaluate
predictions = model.predict(X_test)
mse = mean_squared_error(y_test, predictions)
print(f"Mean Squared Error: {mse}")

# Plot the True values vs Predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions)
plt.xlabel("True Values")
plt.ylabel("Predictions")
```

```
plt.show()
```





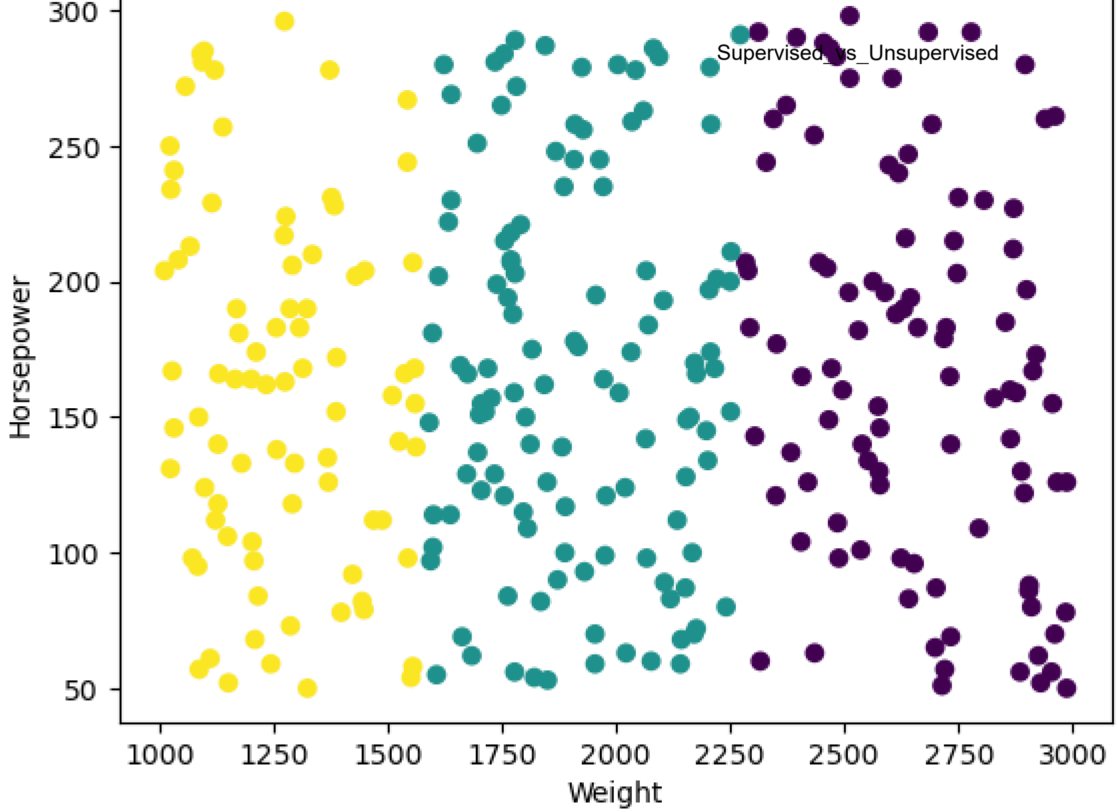

```
Mean Squared Error: 146.02051889440582
```




## Part 2: Vehicle Clustering (Unsupervised Learning)

**Scenario** - You are working for an automotive company, and your task is to cluster vehicles into groups based on their features such as weight, engine size, and horsepower.

In [ ]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import warnings
from sklearn.cluster import KMeans

warnings.filterwarnings('ignore')

# Generate synthetic dataset for vehicles
np.random.seed(0)
data_size = 300
data = {
    'Weight': np.random.randint(1000, 3000, data_size),
    'EngineSize': np.random.uniform(1.0, 4.0, data_size),
    'Horsepower': np.random.randint(50, 300, data_size)
}
df = pd.DataFrame(data)

# No labels are needed for unsupervised learning
X = df

# Perform KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X)

# Plotting the clusters
plt.scatter(df['Weight'], df['Horsepower'], c=kmeans.labels_)
plt.xlabel('Weight')
plt.ylabel('Horsepower')
plt.title('Vehicle Clusters')
plt.show()
```




# Conclusions of the Exercises: Supervised vs Unsupervised Learning

- Exercise 1: Predicting Building Energy Efficiency (Supervised Learning)

In this exercise, we developed a supervised learning model to predict the energy efficiency of buildings based on various features such as wall area, roof area, overall height, and glazing area. We used a Random Forest regression model and evaluated its performance using the Mean Squared Error (MSE). The model demonstrated a reasonable ability to predict energy efficiency values, as reflected in the MSE and the graphical comparison of predicted versus actual values. This exercise shows how supervised learning can be effective for prediction problems when labeled data is available

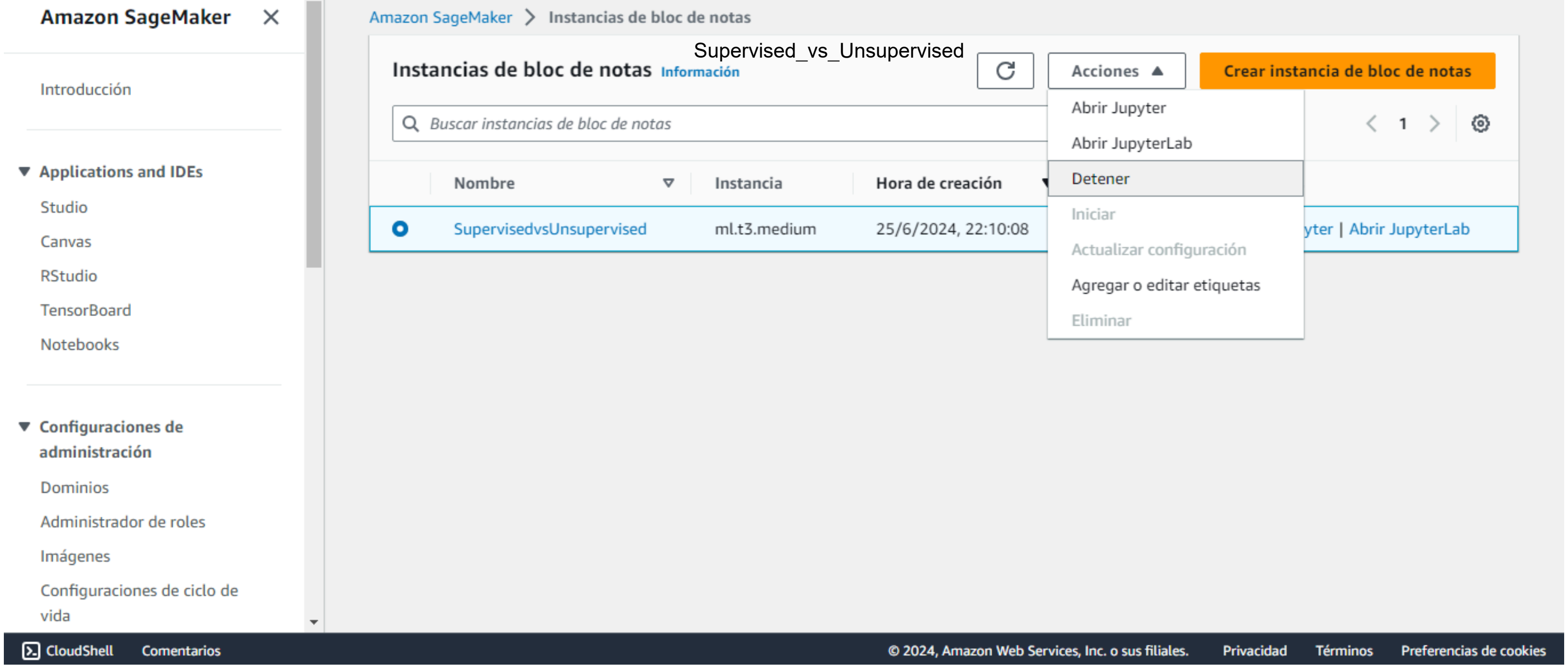
In this exercise, we used the KMeans algorithm to cluster vehicles based on their specifications such as weight, engine size, and horsepower. This unsupervised learning method allowed us to naturally identify three different clusters of vehicles without the need for pre-labeled data. The visualization of the clusters revealed clear groupings based on the provided features. This exercise illustrates how unsupervised learning can be useful for identifying patterns and structures in data when no predefined labels are available.

Both exercises demonstrate how different machine learning approaches can be effectively applied to solve various types of problems, depending on the nature of the available data and the specific objectives of the analysis.

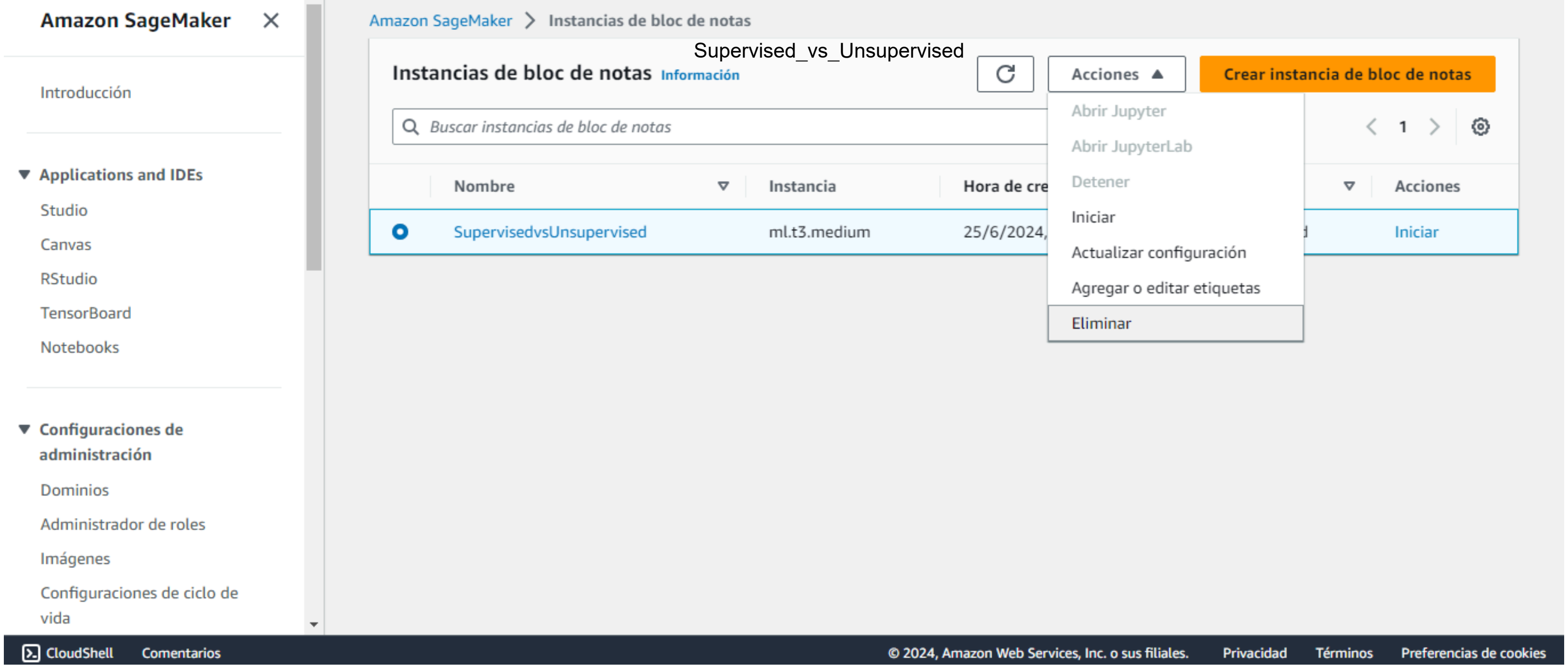
7. **Delete the notebook instance after reviewing the solution on the next page.**

You'll generate synthetic data for both exercises and execute them in an AWS SageMaker Jupyter Notebook instance.

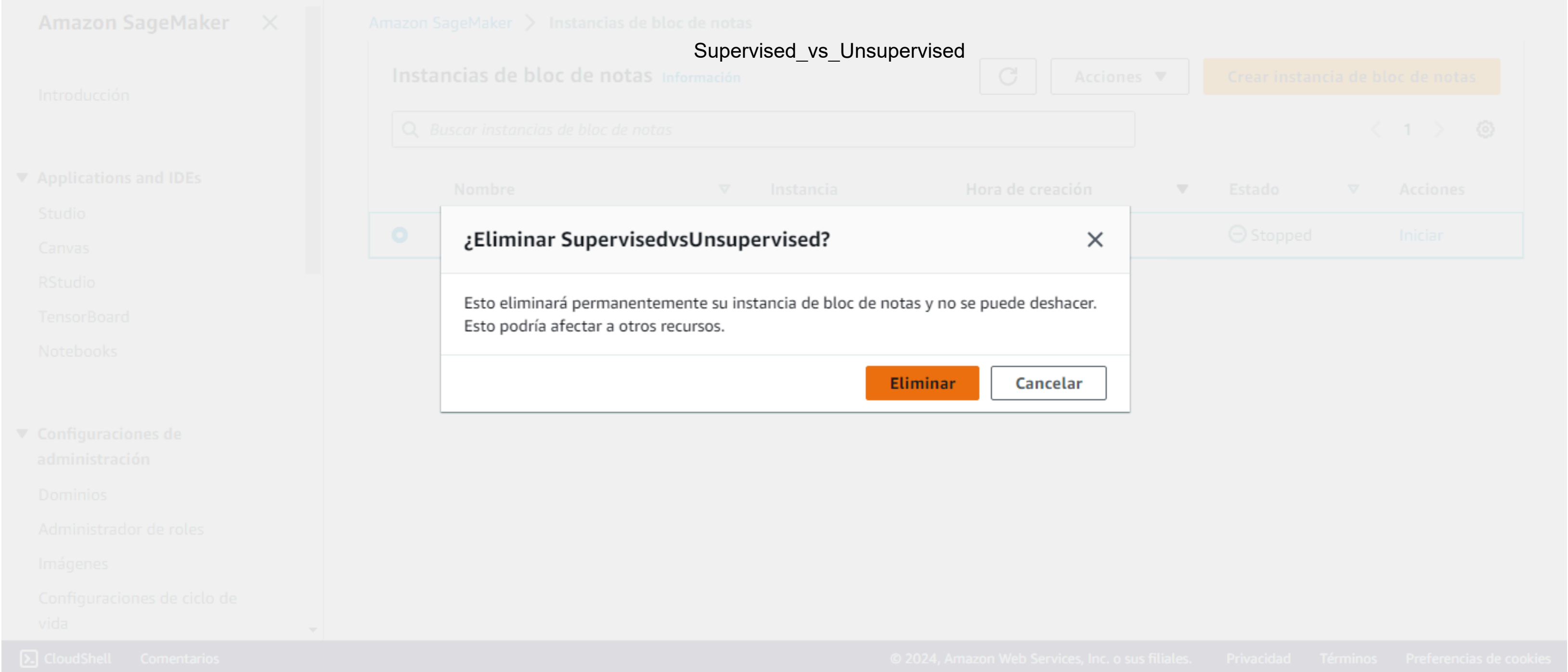
- We wait a moment and click on Actions and "Stop"



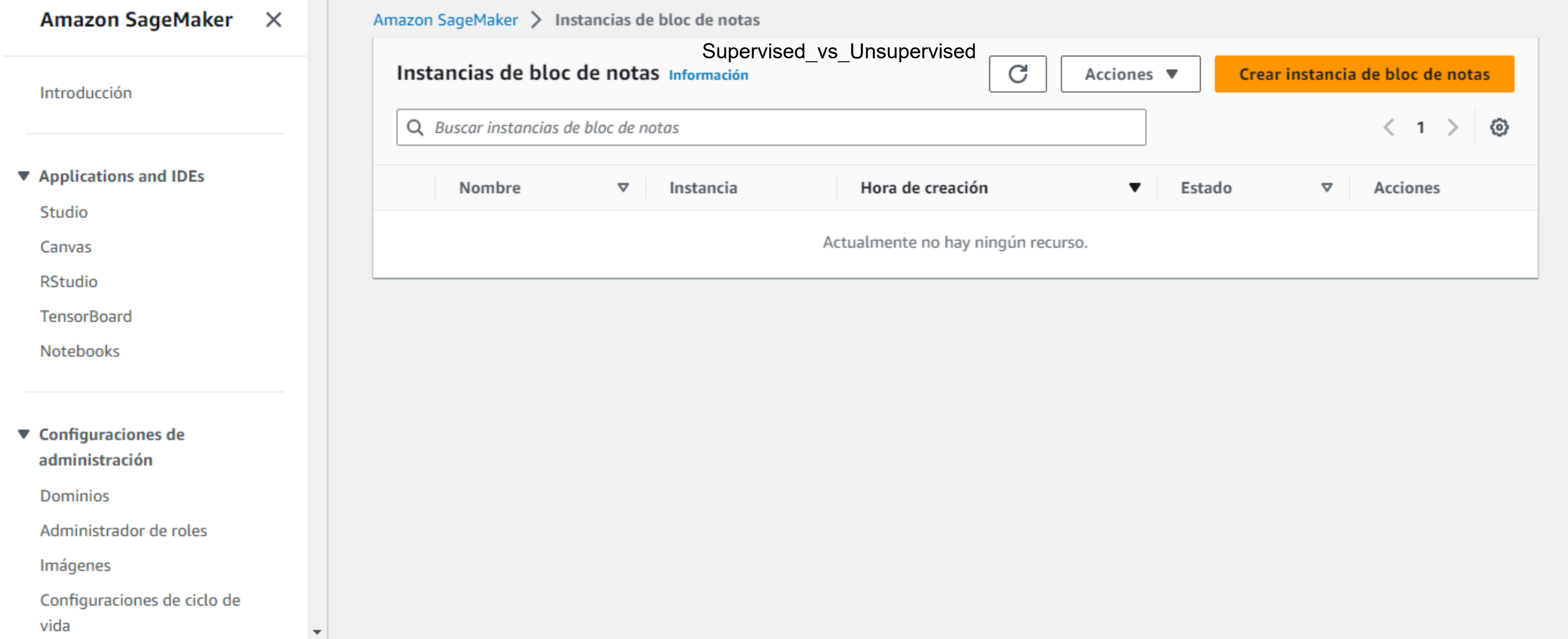

- We wait a moment and click on Actions and "Delete"




- Click on "Delete"




- Check that no instances appear




# Laboratory Completed.

**Note**- These exercises use synthetic data for simplicity and demonstration purposes. In real-world applications, actual data and more complex models might be used. The choice of algorithms - RandomForestRegressor for supervised learning and KMeans for unsupervised learning - is based on their general applicability and ease of understanding for educational purposes.