INTERNET DE LAS COSAS

JORGE ESCAMILLA AMBROSIO

PRACTICA 6

Anomaly Detection Techniques in Python

LOPEZ PEREZ ALBERTO ANDREI

INTRODUCCION

La detección de anomalías, también conocida como detección de valores atípicos o anomalías, es una técnica utilizada en el análisis de datos para identificar observaciones que se desvían significativamente del comportamiento normal o esperado en un conjunto de datos. Estas observaciones inusuales se denominan anomalías o valores atípicos y pueden ser indicativos de eventos raros, errores en los datos, comportamientos anómalos o incluso situaciones sospechosas.

La detección de anomalías tiene aplicaciones en una amplia variedad de campos, como la seguridad de la red, el monitoreo de sistemas, el análisis financiero, la detección de fraudes, el diagnóstico médico, la supervisión industrial y más. El objetivo principal es identificar y resaltar los patrones que se desvían de lo que se considera normal, para que se puedan tomar medidas apropiadas según el contexto.

Existen diferentes enfoques y técnicas utilizadas en la detección de anomalías, y la elección de la técnica adecuada depende del tipo de datos, el dominio de aplicación y los requisitos específicos del problema. A continuación, se presentan algunos de los enfoques más comunes utilizados en la detección de anomalías:

1. Métodos basados en umbrales: Este enfoque implica establecer límites o umbrales en las características o características relevantes de los datos. Cualquier observación que se encuentre fuera de estos límites se considera una anomalía. Este método es simple y directo, pero puede ser limitado cuando los datos presentan variaciones significativas o cuando los patrones de anomalías no son conocidos de antemano.
2. Métodos basados en estadísticas: Estos métodos utilizan técnicas estadísticas para modelar la distribución de los datos y calcular medidas de distancia o desviación entre los valores observados y el modelo estadístico. Si la desviación es significativa, se considera una anomalía. Ejemplos de estos métodos son el cálculo de valores Z, pruebas de hipótesis o estimaciones de densidad.
3. Métodos basados en aprendizaje automático: Estos enfoques utilizan algoritmos de aprendizaje automático para construir modelos que pueden distinguir entre datos normales y anómalos. Los algoritmos pueden ser supervisados, donde se disponen de datos etiquetados con anomalías, o no supervisados, donde el algoritmo debe aprender a partir de datos no etiquetados. Algunos algoritmos comunes incluyen bosques aleatorios, máquinas de soporte vectorial, redes neuronales y métodos de agrupamiento.
4. Métodos basados en series temporales: Estos enfoques se utilizan cuando los datos tienen una estructura temporal y se busca detectar anomalías en función de cambios o desviaciones en el tiempo. Se pueden aplicar técnicas como el suavizado exponencial, modelos ARIMA (Autoregressive Integrated Moving Average) o métodos basados en series temporales.

Métodos basados en clustering: Estos métodos agrupan los datos en diferentes grupos o clusters, y las observaciones que no se ajustan claramente a ningún cluster se consideran anomalías. Los algoritmos de clustering, como el k-means o el DBSCAN, pueden utilizarse para este propósito.

Es importante tener en cuenta que no existe un enfoque universalmente mejor para la detección de anomalías, y la elección de la técnica adecuada depende del contexto y los datos específicos. Además, es fundamental realizar una validación y evaluación cuidadosas de los resultados de la detección de anomalías, ya que los falsos positivos y los falsos negativos pueden tener consecuencias significativas en diferentes aplicaciones.

En Python, hay varias bibliotecas y herramientas disponibles que facilitan la implementación de técnicas de detección de anomalías. A continuación, te presentamos algunas de las bibliotecas más populares y ampliamente utilizadas para la detección de anomalías en Python:

Scikit-learn: Es una biblioteca muy utilizada para el aprendizaje automático en Python. implemente Proporcionaciones de varios algoritmos de detección de anomalías, como el bosque aleatorio (IsolationForest), vecinos más cercanos (Nearest Neighbors), SVM (OneClassSVM) y el método basado en la densidad de Local Outlier Factor (LOF).

PyOD (Python Outlier Detection): Es una biblioteca especializada en la detección de anomalías en Python. Ofrece una amplia gama de algoritmos de detección de anomalías, incluyendo métodos basados en estadísticas, aprendizaje automático y agrupamiento. PyOD también proporciona herramientas para la visualización y evaluación de resultados de detección de anomalías.

AnomalyDetection: Es una biblioteca de Python diseñada específicamente para la detección de anomalías en serie de tiempo. Implementa algoritmos basados en métodos estadísticos como detección de anomalías univariadas, detección de anomalías en secuencias y detección de anomalías recurrentes.

Prophet: Es una biblioteca desarrollada por Facebook para el análisis y pronóstico de series de tiempo. Aunque su objetivo principal es el pronóstico, también puede utilizarse para la detección de anomalías en series de tiempo al modelar tendencias y patrones estacionales.

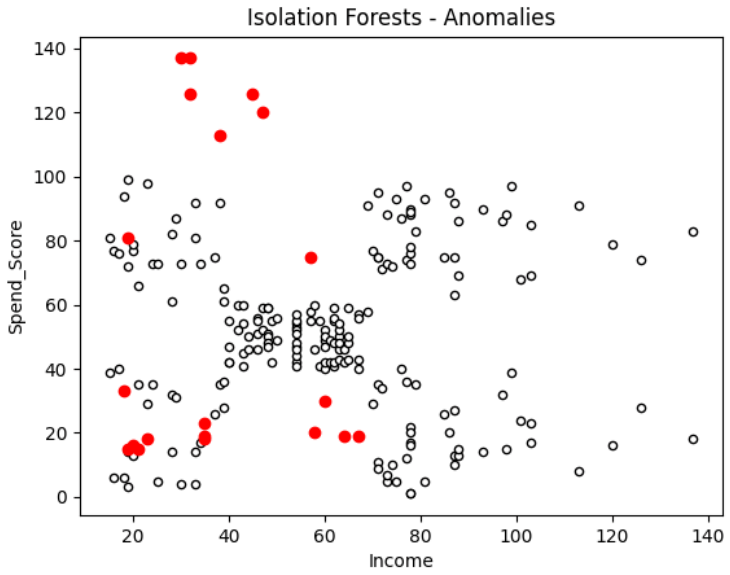
Statsmodels: Es una biblioteca estadística en Python que proporciona herramientas para el análisis y modelado estadístico. Puede utilizarse para realizar pruebas de hipótesis y estimaciones de densidad, que son útiles en la detección de anomalías basadas en métodos estadísticos.

Desarrollando un poco, me di cuenta de que debíamos de poner nuestra ruta en dónde se aprecia la cantidad y el archivo de los customers.

```
# Importing the dataset
data = pd.read_csv('~/Descargas/Mall_Customers.csv')  #,index_col='CustomerID')
#data.head()
```

Para este caso, yo tengo los datos en descargas.

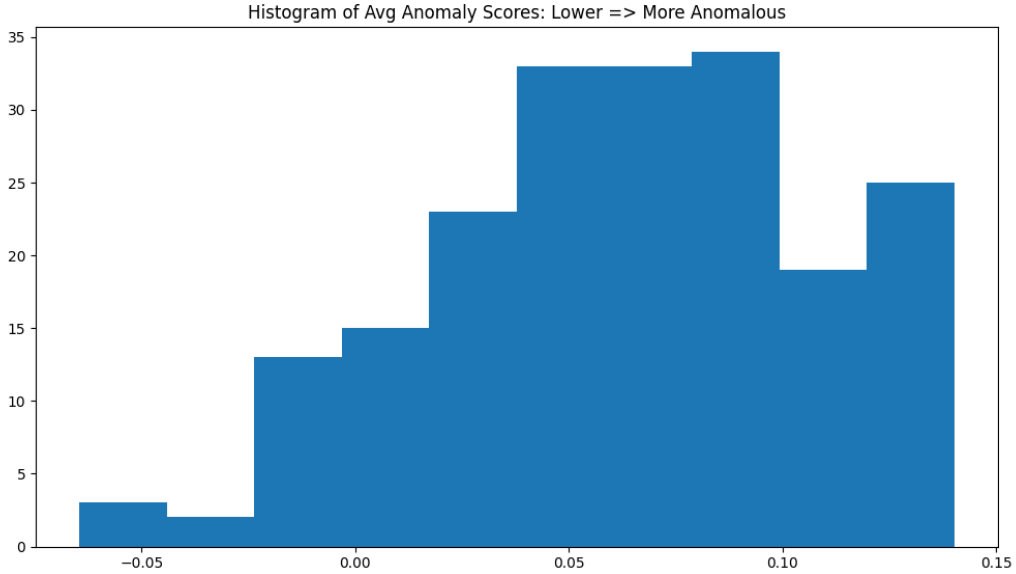
Una vez que hemos instalado algunas bibliotecas que hacían falta, corremos el programa y podemos observar que nos van mostrando imagen por imagen cada detección de anomalías.



Empezamos con Isolation Forests, es un algoritmo de detección de anomalías que utiliza el concepto de árboles de decisión para identificar observaciones inusuales en un conjunto de datos. Fue propuesto por Liu, Ting y Zhou en 2008.

A diferencia de otros algoritmos de detección de anomalías que buscan encontrar patrones normales y luego identificar observaciones que no se ajustan a esos patrones, los Isolation Forests toman un enfoque opuesto. El algoritmo se basa en la idea de que las anomalías son más rápidas de aislar que las observaciones normales en un conjunto de datos.
No requiere una distribución específica de los datos y no hacen suposiciones sobre la forma de los datos normales.
Son capaces de manejar grandes conjuntos de datos de manera eficiente.
Son resistentes a la presencia de atributos irrelevantes y ruido en los datos.
Pueden proporcionar una medida de la importancia relativa de los atributos en la detección de anomalías.

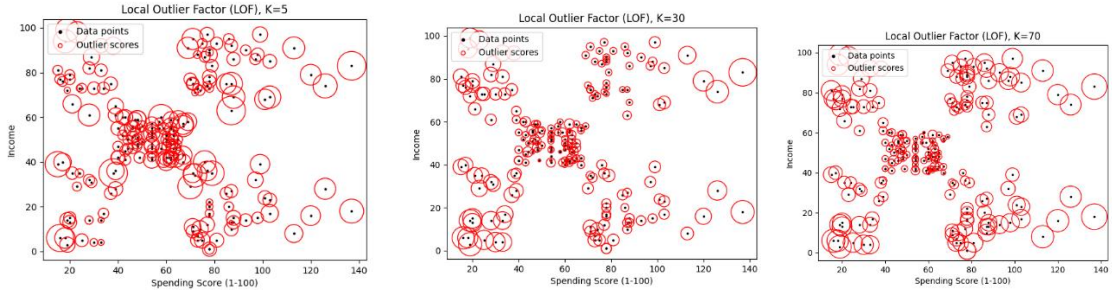
Para la siguiente imagen, tenemos los siguiente:



El histograma muestra la frecuencia de los diferentes rangos de puntajes de anomalías. El eje horizontal representa los rangos de puntajes de anomalías, divididos en intervalos o "bins". El eje vertical muestra la cantidad de observaciones en cada intervalo. Esto permite visualizar cómo se distribuyen los puntajes de anomalías en el conjunto de datos y proporciona información sobre la presencia y magnitud de las anomalías.

Un histograma de los puntajes promedio de anomalías puede ser útil para identificar la presencia de anomalías en un conjunto de datos, así como para comprender la distribución general de los puntajes y establecer umbrales o límites para definir qué se considera una anomalía.

En seguida de esto, vamos a tener tres imágenes que cada una tendrá su detalle:



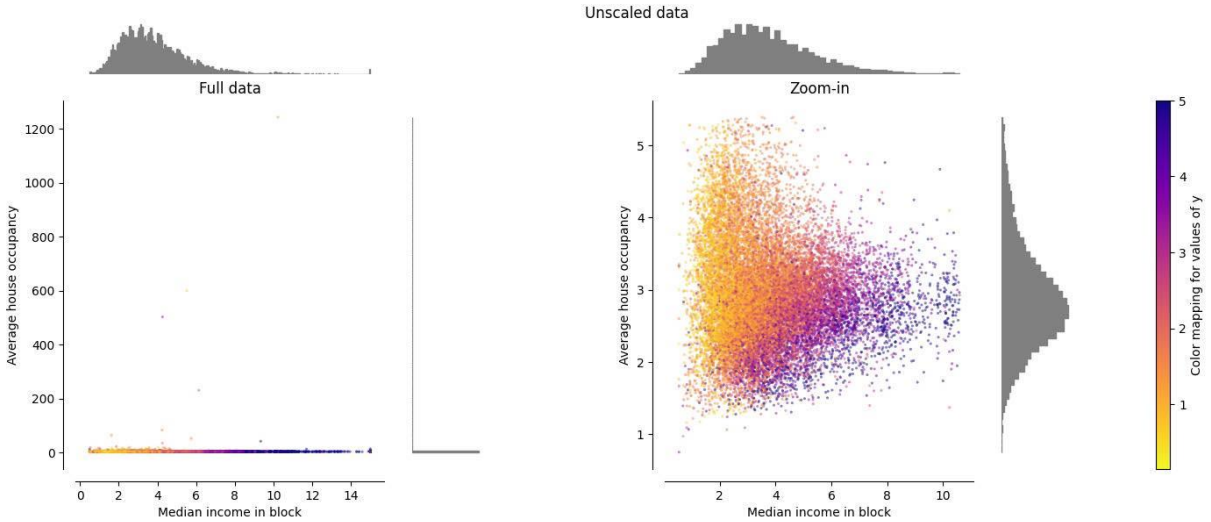
Es un algoritmo de detección de anomalías utilizado para identificar observaciones anómalas en conjuntos de datos. Fue propuesto por Breunig, Kriegel, Ng y Sander en 2000.

A diferencia de los métodos de detección de anomalías globales que buscan encontrar anomalías en todo el conjunto de datos, LOF se basa en la idea de que una observación puede ser considerada anómala en relación con su vecindario local. En lugar de evaluar las desviaciones globales, LOF mide la desviación local de una observación en comparación con sus vecinos cercanos.

El algoritmo LOF asigna un puntaje de LOF a cada observación en el conjunto de datos. Este puntaje indica la desviación de la densidad local de la observación en comparación con la densidad local de sus vecinos. Cuanto mayor sea el puntaje LOF de una observación, más anómala se considera.

LOF (Local Outlier Factor) es un algoritmo de detección de anomalías que mide la desviación local de una observación en relación con la densidad local de sus vecinos. Proporciona una forma efectiva de identificar anomalías en conjuntos de datos considerando la relación con su entorno local.

Para la última imagen, corriendo el programa de ML, tenemos que:



Se refiere a datos que no han sido sometidos a ningún proceso de escala o normalización antes de aplicar algoritmos de detección de anomalías basadas en aprendizaje automático.

En el contexto de la detección de anomalías con algoritmos de aprendizaje automático, es común y recomendable realizar una etapa de preprocesamiento de los datos antes de aplicar el algoritmo. Esto incluye técnicas como la escala y normalización de los datos.

La escala de los datos se refiere a llevar los diferentes atributos o características a una escala similar, generalmente dentro de un rango específico. Esto es importante porque los algoritmos de aprendizaje automático pueden verse afectados negativamente por características con diferentes escalas. Por ejemplo, si un atributo tiene valores en el rango de 0 a 1 y otro atributo tiene valores en el rango de 1000 a 10000, el algoritmo puede dar más importancia al atributo con mayor escala, lo que puede sesgar los resultados de detección de anomalías.

En conclusión, se puede decir que la detección de anomalías en Python es una tarea importante en el análisis de datos que permite identificar observaciones inusuales o atípicas en un conjunto de datos. Python ofrece una variedad de bibliotecas y herramientas que facilitan la implementación de técnicas de detección de anomalías, brindando a los desarrolladores y analistas una amplia gama de opciones para abordar diferentes escenarios y requisitos.

Algunas de las bibliotecas más populares para la detección de anomalías en Python incluyen Scikit-learn, PyOD, AnomalyDetection, Prophet y Statsmodels. Estas bibliotecas fundamentan algoritmos y métodos sofisticados, como Isolation Forests, Local Outlier Factor (LOF), métodos basados en aprendizaje automático y enfoques estadísticos.

La elección de la técnica o el algoritmo adecuado depende del contexto y los datos específicos. Es importante entender las fortalezas y limitaciones de cada método y adaptarlo a las necesidades del problema en cuestión. Además, se recomienda realizar una validación y evaluación cuidadosas de los resultados de la detección de anomalías para minimizar los falsos positivos y falsos negativos, ya que estos pueden tener consecuencias significativas en diferentes aplicaciones.

Python proporciona herramientas para preprocesar y visualizar datos, lo que facilita la exploración y comprensión de los patrones normales y anómalos presentes en los conjuntos de datos. La visualización de los resultados de detección de anomalías puede ser útil para interpretar y comunicar los hallazgos de manera efectiva.