TOP 5 PACKAGES IN MACHINE LEARNING

-Krish Batra (2762856)

- **Scikit-Learn**:
  - **Description**: A versatile and user-friendly library for various machine learning algorithms.
  - **Key Features**: Easy integration with other Python libraries, extensive documentation, and support for preprocessing, classification, regression, clustering, and more.
  - **Use Cases**: General-purpose machine learning tasks, quick prototyping, and educational purposes.

EXAMPLES OF SCIKIT-LEARN USED IN MY PROJECT

https://github.com/disastrousDEVIL/Customer_churn_prediction_ML

```
#now we will create the model
x=df1.drop('Churn',axis='columns')
y=df1['Churn']
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(x,y,test_size=0.2,random_state=5)
```

- **Keras**:
  - **Description**: A high-level neural networks API, written in Python and capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, or Theano.
  - **Key Features**: User-friendly API, modularity, and support for rapid experimentation.
  - **Use Cases**: Quick implementation of neural networks, educational purposes, and small to medium-scale deep learning projects.

- **TensorFlow**:
  - **Description**: An open-source framework developed by Google for machine learning and deep learning applications.
  - **Key Features**: Scalability, flexibility, support for various neural network architectures, and deployment on multiple platforms (mobile, web, servers).
  - **Use Cases**: Deep learning, neural network research, large-scale machine learning models, and production deployment.

EXAMPLE OF TENSOR FLOW AND KERAS USED IN MY PROJECT

https://github.com/disastrousDEVIL/Customer_churn_prediction_ML

```
import tensorflow as tf
from tensorflow import keras
model=keras.Sequential([
    keras.layers.Dense(20,input_shape=(26,),activation='relu'),
    keras.layers.Dense(1,activation='sigmoid')]
)

model.compile(
    optimizer='adam',
    metrics=['accuracy'],
    loss='binary_crossentropy'
)
model.fit(X_train,Y_train,epochs=100)
```

- **OpenCV**

  - **Description**: OpenCV is an open-source library that provides a wide range of algorithms and utilities for computer vision and image processing.
  - **Key Features**:
    - Extensive collection of tools and functions for image processing, computer vision, and machine learning.
    - Support for multiple programming languages, including Python, C++, and Java.
    - Capability to handle real-time image and video processing.
    - Integration with deep learning frameworks like TensorFlow and PyTorch.

Use Cases:

1. **Image Processing**: Tasks like filtering, edge detection, and image transformations.
2. **Computer Vision**: Applications such as object detection, facial recognition, and motion tracking.
3. **Video Analysis**: Real-time video analysis, background subtraction, and video stabilization.
4. **Augmented Reality**: Building AR applications by overlaying information on real-world images.
5. **Machine Learning**: Preprocessing images for training machine learning models and implementing computer vision models.

EXAMPLE OF OPENCV USED IN MY PROJECT

https://github.com/disastrousDEVIL/Cartoon_Filter_ML_K-Means

In [101…

```python
#creating edge mask
def edge_mask(img,line_size,blur_value):
    """
    input:"Gray Scale Image"
    Output:"Edges of Image"
    """

    gray=cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
    gray_blur=cv2.medianBlur(gray,blur_value)
    edges=cv2.adaptiveThreshold(gray_blur,255,cv2.ADAPTIVE_THRESH_MEAN_C,
    cv2.THRESH_BINARY,line_size,blur_value)
    return edges
```

- **Seaborn**

  - **Description**: A Python data visualization library based on Matplotlib, providing a high-level interface for drawing attractive and informative statistical graphics.
  - **Key Features**: Simplifies the creation of complex plots, integrates well with Pandas DataFrames, and provides beautiful default styles and color palettes.
  - **Use Cases**: Exploratory data analysis (EDA), statistical visualizations, and creating complex plots with minimal code.

EXAMPLE OF SEABORN USED IN MY PROJECT

https://github.com/disastrousDEVIL/DigitRecognizerNN

In [24]:
```
import seaborn as sn
plt.figure(figsize=(10,7))
sn.heatmap(cm,annot=True,fmt='d')
plt.xlabel('Predicted')
plt.ylabel('Truth')
```

Out[24]: Text(95.72222222222221, 0.5, 'Truth')