**Tech Titans**

**Jazmine, Yammie, Marcus, Oscar, Mustafa**

**Patricia McManus**

**Animated video of Seth Celtic’s journal click to watch:** [**https://ai.invideo.io/watch/2v3evVdGlGx**](https://ai.invideo.io/watch/2v3evVdGlGx)

**Person Name Seth Celtic-** Seth Celtic is a Bio Machine Learning Engineer whose primary role involves analyzing medical data from lab results and images of tumor cells. He collects statistics related to tumor size and length, leveraging this data to develop predictive models for tumor detection. Specifically, Seth focuses on using thermal images to detect breast cancer tumors

**Task:**

* Create a model that can predict breast cancer tumors
* Day in the life of Seth Celtic: From 5am - 8pm

**A Day in the Life of Seth Celtic: An Image Processing Adventure**

### **Early Morning Routine**

In my role as a data analyst specializing in image processing, my day begins at 5am with a workout to maintain both physical health and mental clarity. After a healthy breakfast, I start my workday at 6am by reviewing the day's brief, which involves creating a computer vision model to predict breast cancer tumors using thermal images. The first task is collecting images from a local folder. These images are part of a labeled dataset of breast cancer samples, including both thermal images and other medical data. I ensure each image is properly formatted and free from duplicates. For example, I use Python libraries such as OpenCV to read the images and Pillow to convert them into a consistent format. This initial step is crucial as it lays the foundation for accurate data analysis.

### **Detailed Data Preprocessing**

The preprocessing phase is particularly critical as it involves several meticulous steps to ensure data quality. Normalizing the pixel values ensures consistency across the dataset, which is crucial for the machine learning model to function correctly. For instance, I normalize pixel values to a range of 0 to 1 using OpenCV’s cv2.normalize function. Handling missing data by filling in gaps with the mean values helps maintain the dataset's integrity and prevents biases that could skew the model's predictions. This is done using pandas in Python, where df.fillna(df.mean(), inplace=True) ensures no missing values disrupt the analysis. This part of the process highlights the importance of attention to detail and a thorough understanding of both the data and the tools used in data preprocessing. It is during this phase that I truly appreciate the power of Python libraries like OpenCV and Pillow in streamlining complex image processing tasks.

### **Ready for Analysis**

By 8am, I had successfully collected and preprocessed the data, making it ready for exploratory data analysis (EDA). This process has reinforced the importance of data quality and the meticulous steps required to prepare datasets for effective machine learning applications. Normalizing values, removing inconsistencies, and ensuring data integrity are crucial for reliable model predictions. Through this work, I have gained valuable insights into the complexities of data preprocessing and the critical role it plays in machine learning. For example, I learned that even small inconsistencies in data formatting can significantly impact the model’s performance, necessitating careful validation of each preprocessing step. I feel accomplished and prepared for the next tasks of the day.

**Dataset Review and Initial Insights**

Upon completing the preprocessing phase, I began a thorough review of the dataset for our breast cancer prediction project by 8 AM. This dataset includes a variety of features, including mean radius, texture, smoothness, and also has characteristics like compactness, concavity, and symmetry—all of which could play a big role in predicting cancerous cell types.

I analyzed the distribution of each feature within the dataset to gauge its variability and assess its significance in differentiating between benign and malignant cell types. This initial exploration not only shed light on the data’s underlying patterns but also set the stage for the forthcoming steps of data preprocessing and feature engineering, ensuring a foundation for our predictive modeling.

**Exploratory Data Analysis and Visualizing Data for Insights**

By 9 a.m., I looked into exploratory data analysis (EDA) using Python’s matplotlib and seaborn libraries. My focus was on visualizing the data to gain insights into its distribution. I found histograms, scatter plots, and box plots particularly useful for understanding the distributions of features such as mean radius and texture across benign and malignant samples. These visualizations helped me identify potential patterns and outliers that could impact our model.

Before proceeding around 10 a.m., I decided to enhance my visualization toolkit by installing OpenCV and Pillow. In my virtual environment, I executed the following commands:

pip install opencv-python

pip install pillow

pip install matplotlib

pip install seaborn

With these new tools at my disposal, I’m excited to explore more advanced image processing techniques that could further aid in our analysis.

As the clock struck 11 a.m., I turned my attention to what I plan to have for lunch and pair plots. These plots illuminated relationships by displaying scatter plots of feature pairs alongside their respective histograms. Through these visualizations, I discerned clusters and trends that differentiate benign from malignant samples. By identifying the most influential features for distinguishing between cancer types, we can optimize our model’s performance and enhance its predictive accuracy.

Continuing through the day and now that lunch has been delivered, I eagerly anticipate applying OpenCV and Pillow to create even more intricate visualizations, revealing deeper insights into our data.

**Feature Engineering and Image Processing Functions**

After an amazing pasta salad for lunch I am now ready for the preliminary steps for this pattern recognition task. I have decided to apply the GaussianBlur function from OpenCV and Canny edge detection to reduce noise. I have done careful research into these algorithms and how they can be applied to this pattern recognition problem introduced in breast cancer detection. While selecting a template it becomes evident to me that Open CV will be used to convert images to grayscale before the edge detection methods can be applied.

Before applying those methods, I did some “theoretical” research on how Image Processing functions work. I found out it is all about converting the images to pixel values and multiplying or convolving them with the very functions we are using with the help of opencv to get the new pixel values meaning “preprocessed images”.

I have checked the images using matplotlib after applying the opencv functions using simple 5 lines of code that goes like this:

sample\_img = images[0]

processed\_sample = preprocess\_image(sample\_img)

plt.imshow(processed\_sample, cmap='gray')

plt.title('Processed Image')

plt.show()

After seeing the successful results I decided to go on a lunch break before selecting a model for the classification problem.

**Models Selection**

Ultimately the model I have selected for this project is a Convolutional Neural Network that will detect and classify patterns for breast cancer detection. This is the code I will use to define the model.

model = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(img\_height, img\_width, 1)), MaxPooling2D((2, 2)),

Conv2D(64, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Conv2D(128, (3, 3), activation='relu'),

MaxPooling2D((2, 2)),

Flatten(),

Dense(128, activation='relu'),

Dropout(0.5),

Dense(1, activation='sigmoid') ]) model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

**Model Training and Evaluation**

After a bit of brainstorming on how to split the data I decided to go with wrong and right patterns for breast cancer detection. I have also become inspired not only in the progress of my work but also by knowing how many lives will be changed for the better with the development of this model. This code will be used to split the data into training and validation sets.

X\_train, X\_val, y\_train, y\_val = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

As I begin training the model things are looking promising however the CNN model does have faults. I will apply some data augmentation methods to mitigate decreasing the performance of the model. The following block of code will be applied.

datagen = ImageDataGenerator( rotation\_range=20, width\_shift\_range=0.2, height\_shift\_range=0.2, shear\_range=0.2, zoom\_range=0.2, horizontal\_flip=True, fill\_mode='nearest' )

Quality of the raw image is crucial for the accuracy and precision of this model. This means to me that Open CV and Pillow are gonna be more important tools than I initially thought. The evaluation of this model will be done by using this code:

val\_loss, val\_accuracy = model.evaluate(X\_val, y\_val, verbose=0) print(f'Validation accuracy: {val\_accuracy\*100:.2f}%')

**Learning and Optimizing SVM Hyperparameters**

In delving into Support Vector Machines (SVMs) and their hyperparameter optimization, several key insights and methodologies have emerged. SVMs are powerful tools for classification and regression tasks, particularly valued for their ability to find optimal decision boundaries even in complex, high-dimensional spaces.

**Hyperparameter Optimization:**

One of the critical tasks in using SVMs effectively is optimizing their hyperparameters.

These parameters include:

* **C:** Controls the trade-off between achieving a low training error and minimizing the norm of the weights.
* **Kernel choice (linear, polynomial, RBF, etc.):** Determines the form of the decision boundary.
* **Kernel-specific parameters:** Such as the degree for polynomial kernels or gamma for RBF kernels.
* **Methods for Optimization:** Several techniques can be employed to find the optimal hyperparameters for SVMs:

**1. Grid Search:** A systematic approach where we define a grid of hyperparameter values and evaluate performance for each combination. While exhaustive, it ensures thorough coverage of the parameter space.

**2. Random Search:**This method randomly selects combinations of hyperparameters and evaluates their performance. It is less computationally intensive than grid search and can sometimes yield better results by exploring less obvious configurations.

**3. Bayesian Optimization:** A probabilistic model-based approach that balances exploration (searching in less-explored areas) and exploitation (focusing on areas likely to yield better performance).

**Practical Considerations:**

During my exploration, it became evident that the choice of optimization method depends on factors such as dataset size, computational resources, and desired model performance. Larger datasets might benefit from random search or Bayesian optimization due to their efficiency, whereas smaller datasets could warrant grid search for a more exhaustive exploration.

At Sunset As the clock nears 8 pm, I pause. What have I learned? Collaboration—the heartbeat of progress. Challenges—the stepping stones to breakthroughs. And in the quiet of dusk, I reflect on the lives we aim to touch—the women whose futures hinge on our pixels.

Seth Celtic, the Data Whisperer, signing off. Tomorrow, we continue our odyssey—one pixel at a time.