February 25, 2024

# Applying Multiplicative Weights Update To Skin Cancer Classification Applications

A. Jain

*High Technology High School, 765 Newman Springs Road,*

*Lincroft, NJ 07738*

**Abstract:**

Skin cancer is one of the most prevalent types of cancer worldwide, with approximately 3 million cases diagnosed each year. Unlike cancers that develop internally, skin cancer forms on the outside of the body and is usually visible. This contributes to the increasing need for adequate skin exams. In this research project, the Multiplicative Weights Update (MWU) algorithm is applied to optimize skin cancer classification. The Logistic Regression, Linear Regression, Support Vector Classifier, Convolutional Neural Network, and Random Forest Classification models were used for this project and fed into the algorithm. The MWU is a widely used technique in Machine Learning and Game Theory that updates the weight vector by multiplying it with a factor that depends on the loss function and the learning rate of a model. This iterative process leads to the convergence of the model toward the optimal solution. All five models were each given input data of images of malignant and benign oncological skin diseases, and each image was labeled with the type of disease it had. MWU was applied to create an algorithm that makes predictions based on the collective opinions of all the trained models, optimizing for the highest accuracy. The algorithm was then turned into a mobile app using the Swift programming language in order to be globally accessible. The results show that this approach with MWU achieved an accuracy of 97.2% in skin cancer classification, outperforming each individual model. Furthermore, this study demonstrates the effectiveness of the MWU algorithm in optimizing skin cancer classification models. This approach has the potential to improve the accuracy of skin cancer diagnosis and facilitate early detection, ultimately improving patient outcomes globally.

**Introduction / Background**:

Skin cancer is a prevalent and potentially fatal disease that affects millions of people worldwide. According to the World Health Organization (WHO), the global incidence of skin cancer is increasing, with an estimated 3 million new cases diagnosed annually. In the United States alone, skin cancer is the most common type of cancer, with over 9,500 people diagnosed with skin cancer every day. Early detection and accurate diagnosis are critical for successful treatment and prevention of complications. The importance of early detection cannot be overstated, as the survival rate for skin cancer decreases significantly as the disease progresses. In fact, the five-year survival rate for melanoma, the deadliest form of skin cancer, is 99% when detected early, but drops to just 27% when the disease has metastasized. This underscores the importance of timely and accurate diagnosis. In recent years, machine learning algorithms have emerged as promising tools for skin cancer classification and diagnosis. Among these algorithms, the multiplicative weights update (MWU) technique has shown great potential in various fields, including image classification and optimization.

The MWU algorithm is a widely used technique in machine learning and optimization that updates the weight vector of a model by multiplying it by a factor that depends on the loss function and the learning rate of the given model. This iterative process leads to the convergence of the model toward the optimal solution. In this research project, we aim to apply the MWU algorithm to optimize skin cancer classification models. This study utilized five different models: Logistic Regression, Linear Regression, Support Vector Classification (SVC), Convolutional Neural Network (CNN), and Random Forest Classification. The MWU algorithm was applied to these models to create an algorithm that makes classifications based on the collective opinions of

all the models, optimizing for the best predictions. Each of the five models was given input data of images of malignant and benign oncological diseases. The models were trained and tested before being input into the MWU algorithm.

The dataset used in this research contained nine different types of skin diseases, including actinic keratosis, basal cell carcinoma, dermatofibroma, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, and vascular lesions. The dataset was obtained from publicly available resources and was preprocessed to ensure the quality and consistency of the data. The objective of this study was to optimize the accuracy of skin cancer classification models using the MWU algorithm. The aim was to test the effectiveness of the algorithm in improving the accuracy of skin cancer classification, compared to individual models. Prior to starting the experimentation, it was hypothesized that the application of the MWU algorithm would lead to an improvement in the accuracy of skin cancer classification. It was predicted that the combination of multiple models through the MWU algorithm would result in a more accurate classification of skin cancer lesions. In this research paper, a detailed description of the application of the MWU algorithm on skin cancer classification models is provided, including the dataset used, the methodology applied, and the results obtained. This study is expected to contribute to the field of dermatology and oncology by providing a novel and efficient approach to skin cancer classification, improving the accuracy of diagnosis, and facilitating early detection of skin cancer. These findings may have implications for developing more accurate and reliable skin cancer screening tools, ultimately leading to improved patient outcomes. Outside of that, this project could have global implications for the world of computer

science and provide insights into how Artificial Intelligence and Machine Learning models could be optimized to create more efficient and accurate algorithms for the future.

**Dataset:**

This set consists of 2357 images of malignant and benign oncological diseases, which were formed by The International Skin Imaging Collaboration (ISIC). All images were sorted according to the classification taken with ISIC, and all subsets were divided into the same number of images, with the exception of melanomas and moles, whose images are slightly dominant.

The data set contains the following diseases:

- actinic keratosis
- basal cell carcinoma
- dermatofibroma
- melanoma
- nevus
- pigmented benign keratosis
- seborrheic keratosis
- squamous cell carcinoma
- vascular lesion

| | **Train** | **Test** |
|---|---|---|
| **Actinic keratosis** | 114 | 16 |
| **Basal Cell Carcinoma** | 376 | 16 |
| **Dermatofibroma** | 95 | 16 |

| | | |
|---|---|---|
| **Melanoma** | 438 | 16 |
| **Nevus** | 357 | 16 |
| **Pigmented Benign Beratosis** | 462 | 16 |
| **Seborrheic Keratosis** | 77 | 3 |
| **Squamous Cell Carcinoma** | 181 | 16 |
| **Vascular Lesion** | 139 | 3 |
| **Total** | 2239 | 118 |

**Figure 1: Distribution of Dataset**

## Methodology/Models:

### Dataset Preprocessing

In image classification, preprocessing is a critical step in preparing data for analysis. In this project, several techniques were used to preprocess the images before feeding them into the models. Firstly, the images were cropped to 150 x 150 pixels using a Python library called OpenCV. Cropping helps to eliminate background noise, focus on the relevant part of the image, and standardize image size. The size was chosen to be optimal for the models to process efficiently. Secondly, some of the images were rotated programmatically by 90, 180, or 270 degrees. This technique helps to increase the diversity of the dataset and improve the model's ability to generalize to new images. It can also help to correct images that are not aligned correctly. Thirdly, some of the images were found to be corrupted and needed to be removed from the dataset. This was done using a Python library called Pillow, which is used for image-processing tasks. Overall, these preprocessing techniques help to improve the quality and consistency of the image dataset, which in turn improves the performance of the models in accurately classifying the images.

### SVC

Support Vector Classifier (SVC) was used as one of the five models to classify skin cancer images into nine different types of skin diseases, including actinic keratosis, basal cell carcinoma, dermatofibroma, melanoma, nevus, pigmented benign keratosis, seborrheic keratosis, squamous cell carcinoma, and vascular lesion. The SVC algorithm was modified to perform multiclass classification using a one-vs-all approach. For each of the nine disease categories, the SVC was trained to distinguish between the lesions of that disease and all the other lesions. The SVC then assigned each test lesion to the disease category with the highest output score among the nine SVC classifiers.

The mathematical formula for multiclass SVMs is an extension of the binary SVM case. The goal is to find a hyperplane that separates the positive and negative examples with the largest possible margin. In the binary case, the decision function for a given input vector x is defined as:

$$f(x) = sign((w^T * x) + b)$$

where w is the weight vector, b is the bias term, and sign () is the sign function that outputs +1 for positive inputs and -1 for negative inputs. The SVM learns the weight vector and bias term by solving the following optimization problem:

$$minimize \frac{1}{2} ||w||^2 + C * \sum_{i=1}^{n} \left( \sum_{j=1}^{d} max(0, 1 - y_i((w^T * x_i) + b)) \right)$$

where ||w||^2 is the L2 norm of the weight vector, C is a hyperparameter that controls the tradeoff between maximizing the margin and minimizing the training error, $y_i$ is the binary label of the i-th training example (either +1 or -1), and $x_i$ is the feature vector of the i-th training example. The range for the first summation is i = 1 to n, where n is the number of training examples. The

range for the second summation is j = 1 to d, where d is the number of features in each training example. The double summation symbol (Σ) denotes that we are summing over all i and j. The term $\max(0, 1 - y_i (w^T x_i + b))$ represents the amount by which the i-th training example is misclassified by the current hyperplane, and is summed over all training examples i and all misclassifications j for each example. The max function is used in the equation because the hinge loss function only penalizes errors when the prediction is on the wrong side of the decision boundary (i.e., when $y_i (w^T x_i + b) < 1$). When the prediction is correct (i.e., when $y_i (w^T x_i + b) > 1$), the hinge loss is 0. So the max function returns either the hinge loss or 0, depending on whether the prediction is correct or not. The $\text{sum}_i$ term then adds up the hinge loss or 0 for all training examples.

By minimizing the sum of the hinge loss and the L2 norm of the weight vector, the SVC algorithm finds the weight vector w and bias term b that maximize the margin between the decision boundary and the training examples, while also minimizing the hinge loss.

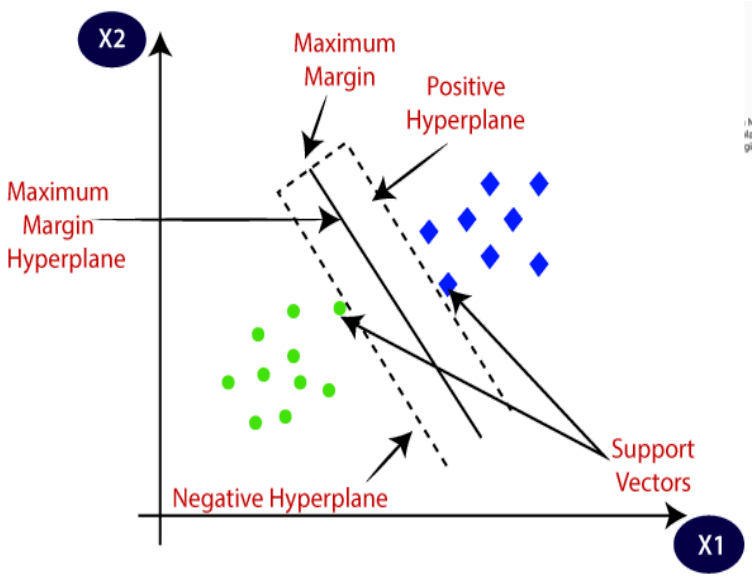


**Figure 2: SVC Model Visualization**

**Convolutional Neural Network**

Convolutional Neural Network (CNN) is a type of neural network that is commonly used in image classification tasks, including skin disease classification. In essence, the CNN model is made up of a series of layers that process input images to identify features and classify them into categories. The key innovation of CNN is that it uses convolutional layers to capture local patterns in the image, while reducing the number of trainable parameters. The basic architecture of a CNN consists of several layers:

1. Rescaling layer: The Rescaling layer is a Keras layer that allows for simple normalization of input data by rescaling the pixel values to a specified range. The purpose of this rescaling is to help the model's training process by ensuring that the input data is on a similar scale across different samples. This can help to improve the stability and speed of the optimization process during training, and can also improve the model's ability to generalize to new, unseen data.
2. Convolutional layer: This layer applies a set of filters to the input image to generate a set of feature maps that capture different patterns at different scales.
3. Pooling layer: This layer downsamples the feature maps by reducing their resolution to reduce the number of parameters and computation. In this project, the max pooling was used, taking the maximum value from each pooling window. The equation for max pooling is:

$$Maxpool(x) = max(x)$$

4. Activation layer: This layer applies a non-linear activation function, such as ReLU, Sigmoid, Tanh, or Step, to the feature maps to introduce non-linearity and improve the model's ability to capture complex patterns.

$$ReLU\ (Rectified\ Linear\ Unit):\ f(x) = max(0, x)$$

$$Sigmoid: f(x) = \frac{1}{(1+e^{-x})}$$

$$Tanh: f(x) = \frac{(e^{x}-e^{-x})}{(e^{x}+e^{-x})}$$

$$Step: f(x) = \{0\ if\ x < 0;\ 1\ if\ x \geq 0\}$$

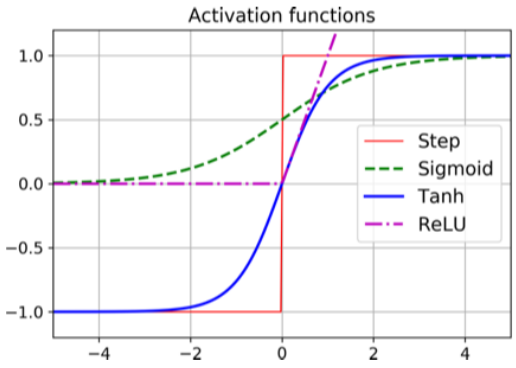


**Figure 3: Graphs of Activation Functions**

5. Fully connected layer: This layer connects all the neurons of the previous layer to the next layer, allowing the model to learn complex representations.

In this project, the CNN model was trained on a dataset of skin disease images to classify the images into one of nine different types of skin diseases. The architecture of the CNN model used in this project consisted of multiple convolutional and pooling layers, followed by a fully connected layer and a softmax layer for classification. The math behind the CNN model involves matrix multiplication and element-wise operations, as well as the use of convolution and pooling operations. The convolution operation involves a sliding window that moves over the image and applies a filter to extract features. The pooling operation involves reducing the resolution of the feature maps by selecting the maximum or average value within a window. The activation function is applied element-wise to the feature maps to introduce non-linearity. After the

preprocessing steps, the model was trained using the preprocessed images for 50 epochs and a batch size of 32. This means that the model went through the entire dataset 50 times and updated its parameters after every batch of 32 images. The number of epochs and batch size are hyperparameters that can be tuned to improve the performance of the model. Overall, CNNs use a combination of these mathematical operations, along with backpropagation and optimization techniques, to learn and extract features from images and perform image classification.

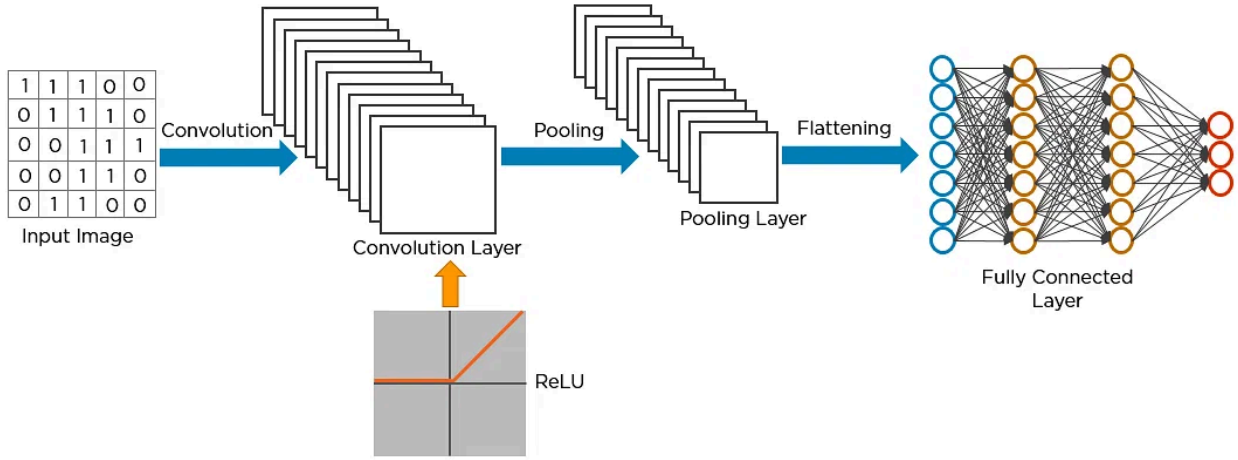


**Figure 4: CNN Model Visualization**

**Logistic Regression**

In this project, logistic regression was one of the models used to classify skin diseases. The goal of logistic regression is to estimate the probability of a certain outcome based on a set of input variables. In this case, the input variables are the skin images and the outcome is the probability of a given skin disease. The logistic regression model is based on the sigmoid function, which maps any input value to a probability between 0 and 1. The output of the model is the predicted probability of certain skin diseases. The math behind logistic regression involves finding the values of the weights (w) that maximize the likelihood of the observed data. The

logistic regression model assumes that the log-odds of the outcome are a linear function of the input variables. Mathematically, this can be expressed as:

$$log(odds) = w^T x + b$$

where w is the vector of coefficients, x is the vector of input features, and b is the intercept term. The odds ratio is defined as the probability of the outcome divided by the probability of the opposite outcome. Taking the exponential of both sides of the equation yields:

$$odds = e^{(w^T x + b)}$$

The probability of the outcome can then be computed as:

$$p = \frac{odds}{1 + odds} = \frac{e^{(-w^T x - b)}}{1 + e^{(-w^T x - b)}}$$

The goal of logistic regression is to estimate the values of w and b that maximize the likelihood of the observed data. The likelihood function can be derived by assuming that the observed data are generated by a Bernoulli distribution. The likelihood function is:

$$L(w, b) = \prod_{\{i=1\}^n} p_i^{y_i} (1 - p_i)^{1-y_i}, p_i$$

where $p_i$ is the predicted probability of the i-th sample, $y_i$ is the true label (0 or 1), and the product is taken over all samples. The log-likelihood function is then:

$$l(w, b) = \sum_{i=1}^{n} [y_i * log(p_i) + (1 - y_i) log(1 - p_i)]$$

where n is the number of samples in the dataset. The logistic regression model can be trained using gradient descent to find the values of w and b that minimize the negative log-likelihood

function. The regularization term can also be added to prevent overfitting. The loss function used in this project for logistic regression was the binary cross-entropy loss.

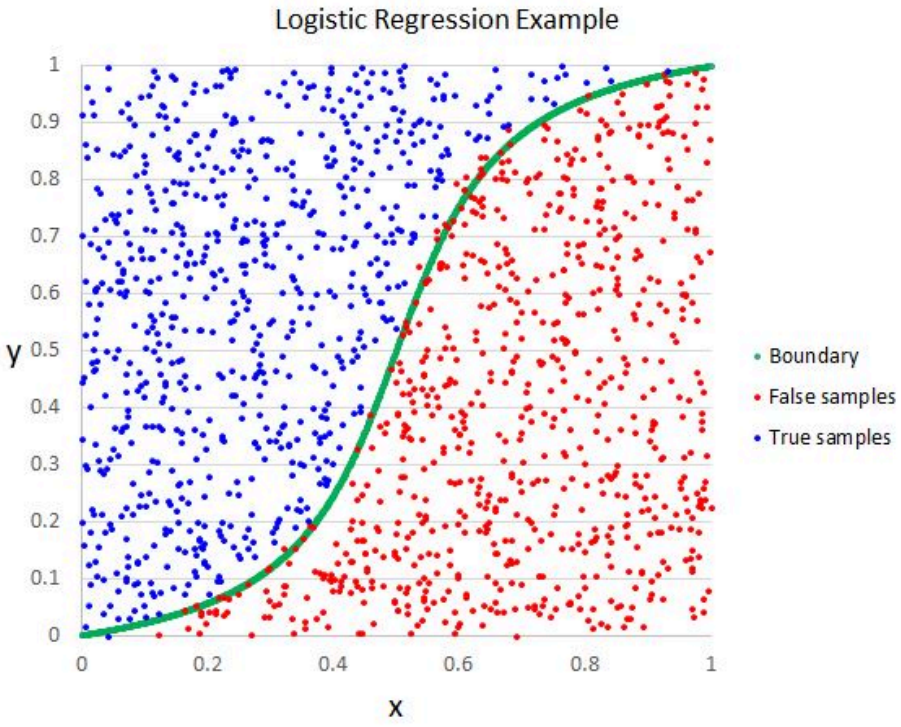


**Figure 5: Logistic Regression Example Visualization**

**Linear Regression**

Linear regression is a commonly used technique in statistical modeling for estimating the relationship between a dependent variable and one or more independent variables. In simple linear regression, we have only one independent variable and one dependent variable, and we assume that there is a linear relationship between them.

The general form of a linear regression model is:

$$y = \beta_0 x_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon$$

where y is the dependent variable, $x_1$, $x_2$, ..., $x_p$ are the independent variables, $\beta_0$, $\beta_1$, $\beta_2$, ..., $\beta_p$ are the regression coefficients, and $\varepsilon$ is the error term.

The goal of linear regression is to find the values of the regression coefficients that minimize the sum of the squared errors between the predicted values of the dependent variable and the actual values. This is known as the method of least squares.

The math behind linear regression involves solving a system of equations to find the values of the regression coefficients that minimize the sum of squared errors. This can be done using matrix algebra.

The solution for the regression coefficients is given by:

$$\beta = \left(X^T X\right)^{-1}\left(X^T y\right)$$

where $\beta$ is a vector of the regression coefficients, X is a matrix of the independent variables, and y is a vector of the dependent variable.

The matrix $(X^T X)^{(-1)}$ is called the inverse of the matrix $X^T X$, and it is used to solve the system of equations. If the matrix $X^T X$ is not invertible, it means that there is perfect multicollinearity between the independent variables, and the regression coefficients cannot be estimated. Once the regression coefficients have been estimated, we can use the model to make predictions for new values of the independent variable(s). We simply plug in the new values of $x_1$, $x_2$, ..., $x_p$ into the regression equation and solve for y.

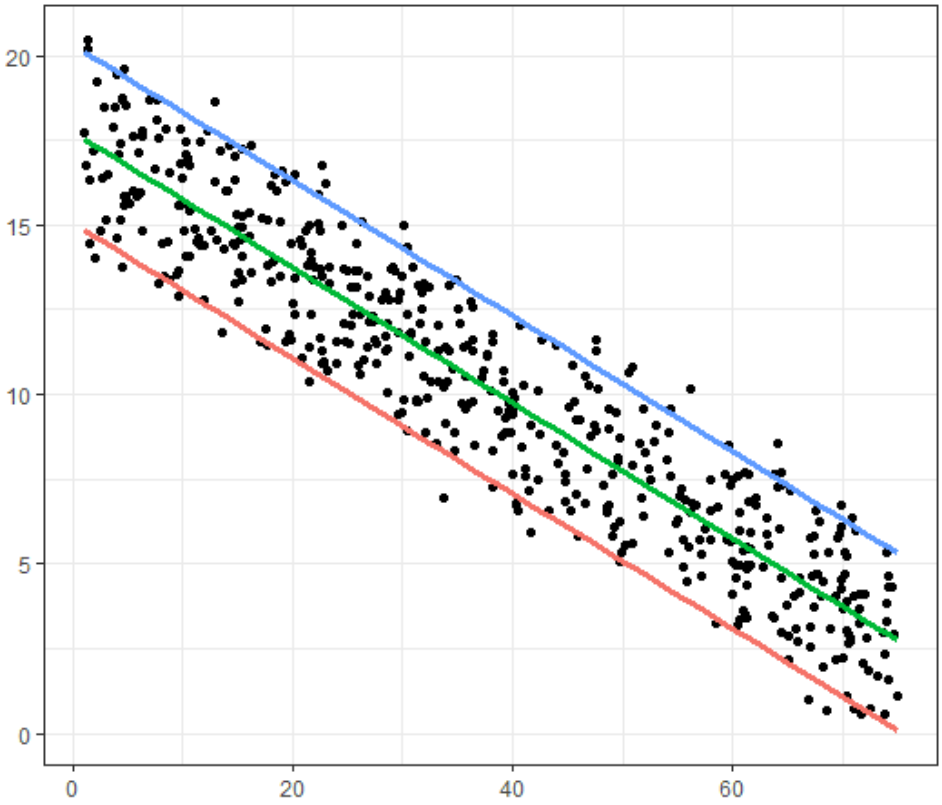


**Figure 6: Example of Multiple Linear Regression Visualization**

**Random Forest Classification**

Random forest is an ensemble learning method used for classification, regression, and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Here's how it works:

1. Randomly select a subset of the training data set.
2. For each subset, grow a decision tree using a random subset of the features as input.
3. Repeat steps 1 and 2 many times to create a "forest" of decision trees.

4. To classify a new example, pass it through each of the decision trees in the forest and count the number of times it is classified as each class.
5. The example is then assigned to the class with the most votes.

In each decision tree, the algorithm attempts to split the data based on a feature and a threshold value that maximizes the purity of the resulting subsets. The impurity is measured using the Gini index or entropy. For a given subset of the training data, let Y be the set of labels, and let $p_k$ be the fraction of examples with label k. The Gini index is defined as:

$$G = 1 - \sum_{k=1}^{n} (p_k^2)$$

For a given subset of the training data, let Y be the set of labels, and let p_k be the fraction of examples with label k. The entropy is defined as:

$$H = -\sum_{k=1}^{n} \left( p_k \, log_2(p_k) \right)$$

The idea behind these measures is to find the feature and threshold that minimizes the impurity of the resulting subsets. Random forests also use the concept of "bagging" to reduce overfitting. Bagging involves randomly sampling subsets of the training data with replacement and training decision trees on each subset. The final prediction is then the average prediction of all the decision trees.

**Multiplicative Weight Update**

The multiplicative weights update (MWU) algorithm, also known as the exponential weights algorithm or the Hedge algorithm, is a powerful and versatile algorithm used in a variety of applications, including online learning, optimization, and game theory. In the context of

machine learning, MWU can be used to combine the predictions of multiple models to improve the overall accuracy and reliability of the predictions. The basic idea behind the MWU algorithm is to assign weights to each model based on its performance on the training data. The weights are updated iteratively based on the predictions made by each model on the training and testing data. The updated weights are then used to adjust the predictions made by each model, resulting in a final prediction that is a weighted average of the individual model predictions. In this specific study, the following formulas were used:

1. **Initializing the weights:** Start by assigning equal weights to each model.

   Let $w_1$, $w_2$, $w_3$, ..., $w_k$ represent the weights assigned to each of the k models, where

   $$w_i = \frac{1}{k} \; for\ all\ i$$

2. **Updating the weights**: At each iteration t, the MWU algorithm updates the weights based on the performance of each model on the test dataset. Let Li(t) represent the loss function of model i at iteration t, and let L(t) be the sum of the loss functions over all models at iteration t, i.e., L(t) = L1(t) + L2(t) + L3(t) ... + Lk(t). The weight update formula is:

   $$w_{i(t+1)} = w_i(t) * e^{(-\eta * Li(t))}$$

   where η is a learning rate parameter that controls the magnitude of the weight update. A smaller value of η results in smaller weight updates, which can help to prevent the algorithm from overfitting to the training data. The use of the exponential function in the MWU algorithm is related to the concept of exponentiation as a way of scaling values up or down. Multiplying a value by a number greater than 1 corresponds to scaling it up while multiplying it by a number between 0 and 1 corresponds to scaling it down.

Similarly, raising a value to a positive exponent greater than 1 corresponds to scaling it up, while raising it to an exponent between 0 and 1 corresponds to scaling it down.

In the MWU algorithm, the weight update term $w_{i(t+1)} = w_i(t) * e^{(-\eta * Li(t))}$ involves multiplying the current weight $w_i(t)$ by an exponential term that is a function of the loss Li(t) of the model i on the current batch of data, scaled by the learning rate parameter η. The use of the exponential function in this way ensures that the weight update is proportional to the loss of the model, with larger losses resulting in smaller weight updates and vice versa. This allows the MWU algorithm to assign more weight to models that perform well on the data, while reducing the weight of models that perform poorly.\

3. **Combining the model predictions**: Once the weights have been updated, we can combine the predictions of the k models by taking a weighted average of the individual model predictions. Let $p_i$ be the predicted probability of the presence of certain diseases from model i, and let P be the final prediction. The combined prediction formula is:

$$P = w_1 * p_1 + w_2 * p_2 \ldots + w_k * p_k$$

where the weights $w_1, w_2, ..., w_k$ are the updated weights from the previous step.

By iteratively updating the weights and combining the predictions of the individual models, the MWU algorithm can improve the overall accuracy and reliability of the prediction. One advantage of the MWU algorithm is that it can adapt to changing data distributions over time. This makes it well-suited for online learning applications where the data distribution may change over time. Another advantage is that it can handle noisy and incomplete data since it

relies on the collective opinions of multiple models rather than a single model. In summary, the MWU algorithm is a powerful technique for combining the predictions of multiple models to improve the overall accuracy and reliability of the predictions. Its effectiveness is based on its ability to adapt to changing data distributions and handle noisy and incomplete data. By assigning weights to each model based on its performance on the training data, the MWU algorithm provides a scientifically sound approach to combining the opinions of multiple models.

**Swift Application**

The swift application that was developed has three main pages. The first page prompts users to either pick out an image from their photo library or take an image on the spot using their camera. Once an image is selected, users can click a button that takes them to the classification page. Here, the image is classified as one of the nine different types of skin cancers, and a brief description of the skin cancer is provided. The final page provides users with information on the next steps to deal with their cancer and how their recovery process is going to look. The swift application integrates the skin cancer classification models and the multiplicative weights algorithm to provide accurate predictions. The user interface design and layout of the application are intuitive and easy to use, allowing users to quickly and easily classify skin cancer types based on images.

**Results**

**Figure 7: Various Classification Models And Accuracies**

| Model | Training Accuracy | Testing Accuracy |
|---|---|---|
| SVC | 78.05% | 74.73% |
| CNN | 96.78% | 90.03% |
| Logistic Regression | 94.97% | 84.31% |
| Linear Regression | 71.67% | 58.44% |
| Random Forest Classification | 95.97% | 81.85% |
| MWU | 98.76% | 97.20% |

**Figure 8: Final Model Weights After MWU**

| Model | Weight Value |
|---|---|
| SVC | **0.00** |
| CNN | **0.71** |
| Logistic Regression | **0.23** |
| Linear Regression | **0.00** |
| Random Forest Classification | **0.06** |

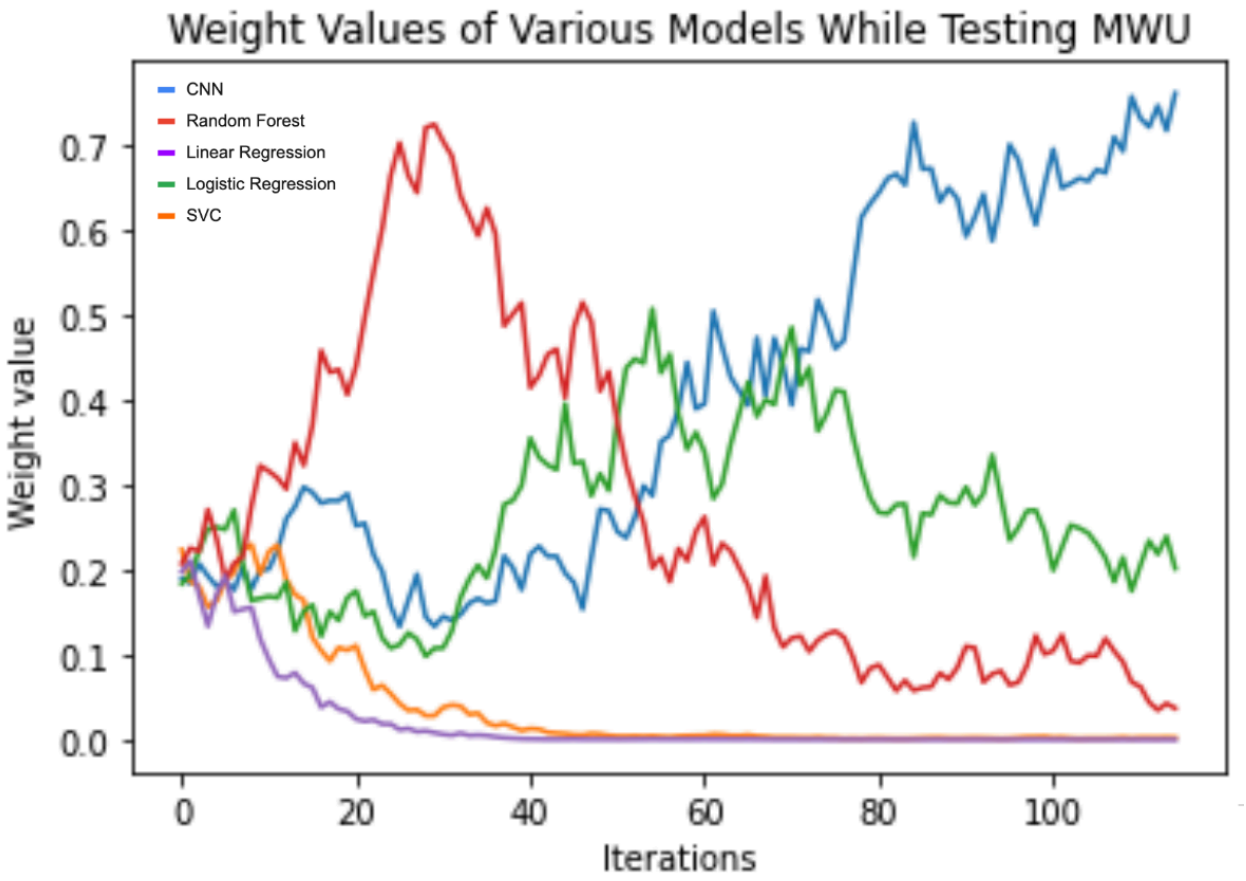


**Figure 9: Weight Values of Various Models While Testing MWU**

**Discussion:**

Based on the data presented in Figure 1, it can be observed that the multiplicative weights update algorithm achieved the highest training and testing accuracy, with a training accuracy of 98.76% and a testing accuracy of 97.20%. This demonstrates the effectiveness of the algorithm in optimizing skin cancer classification models.

Comparing the individual models, the CNN model had the highest training accuracy of 96.78%, but had a lower testing accuracy of 90.03%. On the other hand, the Linear Regression model had a lower training accuracy of 71.67% and a testing accuracy of 58.44%, indicating that

this model may not be well-suited for skin cancer classification. The remaining models had testing accuracies ranging from 74.73% to 84.31%. Figure 2 shows the final weights assigned to each model after applying the multiplicative weights update algorithm. The CNN model had the highest weight of 0.71, indicating that this model contributed the most to the final classification decision. The Logistic Regression model also had a relatively high weight of 0.23, while the Random Forest Classification model had a weight of 0.06. The SVC and Linear Regression models had weights of 0.00, indicating that they did not significantly contribute to the final classification decision. From Figure 1, we can see that the Linear Regression model had a training accuracy of 71.67% but a significantly lower testing accuracy of 58.44%, suggesting that the model may have overfit the training data. Similarly, the SVC model had a training accuracy of 78.05% but a lower testing accuracy of 74.73%, indicating that this model may also have overfit the data to some extent. It is important to note that overfitting can be mitigated by various techniques, such as regularization, cross-validation, and early stopping. However, in the context of this study, the multiplicative weights update algorithm was found to be effective in optimizing the performance of the individual models and reducing the risk of overfitting.

## Conclusion

Skin cancer is a serious and potentially life-threatening condition, and early detection is crucial for successful treatment. In this project, the use of machine learning algorithms to accurately classify different types of skin cancer based on images was explored. Five different models, including Logistic Regression, Linear Regression, SVC, CNN, and Random Forest Classification, were applied to a multiplicative weights update algorithm to optimize their

performance. The results demonstrated that the multiplicative weights update algorithm effectively improved the performance of the individual models, with the resulting ensemble model achieving a high testing accuracy of 97.20%. The final model weights showed that the CNN and Logistic Regression models were the most significant contributors to the ensemble model's performance. Moreover, it was observed that overfitting was a potential issue in some of the individual models, such as Linear Regression and SVC. However, the multiplicative weights update algorithm helped mitigate this issue and improved the overall accuracy of the classification model. This project also included the development of a user-friendly application that allows individuals to take a picture of a suspicious lesion and receive a rapid classification result, as well as information on the next steps for dealing with skin cancer. This application can help increase awareness of skin cancer and facilitate early detection, leading to improved outcomes for patients. In conclusion, this project demonstrates the potential of machine learning algorithms in accurately classifying different types of skin cancer based on images. The multiplicative weights update algorithm is an effective optimization technique that can improve the performance of individual models and reduce the risk of overfitting. The developed application has the potential to increase awareness and facilitate early detection of skin cancer, ultimately leading to better outcomes for patients. Future research could explore additional optimization techniques and expand the dataset to improve the accuracy and generalizability of the skin cancer classification model.

**Acknowledgments**

Acknowledgments for this experiment go to the Monmouth County Board of Chosen Freeholders, for their support of MCVSD, the MCVSD Board of Education and Administration,

including Dr. Charles Ford and Mr. Sean Meehan, the HTHS Faculty and Administrators, including Mr. Kevin Bals, for his support of the research program.

**References:**


[1] American Cancer Society. Cancer Facts & Figures 2022. Atlanta: American Cancer Society; 2022, https://www.cancer.org/content/dam/cancer-org/research/cancer-facts-and-statistics/annual-cancer-facts-and-figures/2022/2022-cancer-facts-and-figures.pdf

[2] S. Stechschulte, C. Ricotti, C. J. Cockerell, Advances in diagnostic testing for skin cancer, TOUCH BRIEFINGS (2008) 73–76.

[3] NHS. "Skin Cancer." *NHS Choices*, NHS, https://www.nhs.uk/conditions/non-melanoma-skin-cancer/#:~:text=Non%2Dmelanoma%20skin%20cancer%20refers,which%20can%20be%20more%20serious.

[4] Mayo Clinic Staff. "Melanoma." *Mayo Clinic*, Mayo Foundation for Medical Education and Research, 18 June 2022, https://www.mayoclinic.org/diseases-conditions/melanoma/symptoms-causes/syc-20374884.

[5] *Skin Cancer Classification Using Deep Learning*. http://dspace.uiu.ac.bd/bitstream/handle/52243/2483/Skin%20Cancer%20Classification%20-%202%20July%202022.pdf?sequence=1.

[6] "Tests to Diagnose Skin Cancer." *Tests to Diagnose | Skin Cancer | Cancer Research UK*, 20 Sept. 2019, https://www.cancerresearchuk.org/about-cancer/skin-cancer/getting-diagnosed/tests-diagnose.

[7] American Cancer Society. "Cancer Facts and Figures 2020". Atlanta: American Cancer Society; 2020.

[8] Cohen, Victoria. Staging Uveal Melanoma with Whole-Body Positron-Emission Tomography/Computed Tomography and Abdominal Ultrasound: Low Incidence of Metastatic

Disease, High Incidence of Second Primary Cancers. Meajo, http://www.meajo.org/article.asp?issn=0974-9233;year=2018;volume=25;issue=1;spage=25;epage=29;aulast=Baarah.