ENGS 106 Final Project Writeup, Winter 2025

Members:

- Sie Hendrata Dharmawan
- Brendan Liu
- Joseph Martinez

Table Of Contents

Comparative Analysis of Obesity Prediction Models: Linear Regression, SVM, and MLP

Overview

This project explores the application of machine learning techniques to predict obesity levels based on eating habits and physical condition data. We implemented and compared three machine learning approaches: Linear Regression, Support Vector Machines (SVM), and Multi-Layer Perceptron (MLP) neural networks to classify individuals into seven obesity categories. Our research aims to determine which predictive model performs most effectively in estimating obesity levels based on lifestyle factors, potentially offering insights for targeted intervention strategies in public health.

Datasets

The UC Irvine Machine Learning Repository published a dataset containing eating habits and physical condition data from individuals in Mexico, Peru, and Colombia. The dataset contains 2,111 individuals and 17 features, including weight, height, and family history. It classifies individuals into seven categories, ranging from "Insufficient Weight" to "Obesity Type III."

The complete dataset can be downloaded from https://www.kaggle.com/datasets/ruchikakumbhar/obesity-prediction (also shown in the notebook)

Features and Transformations

These are the features in the data set:

- Gender. We transform it into 0 for female, 1 for male
- The following features are boolean values. We transform them into 0 for no and 1 for yes
    - Family history: Has a family member suffered or suffers from being overweight?
    - FAVC: Do you eat high caloric food frequently?
    - SMOKE: Do you smoke?
    - SCC: Do you monitor the calories consumed daily?
- The following features are “frequency” values. We transform them into 0 for no, 1 for sometimes, 2 for frequently, and 3 for always. These are ordinal variables.
    - CAEC: Do you eat any food between meals?
    - CALC: How often do you drink alcohol?

- This feature is a categorical variable: MTRANS. The main mode of transportation taken by the individual. We transform it to 0: public transportation, 1: walking, 2: automobile, 3: motorbike, 4: bike.
- Target value: obesity. As mentioned above, we transform it as follows:
    - 0: insufficient weight
    - 1: normal weight
    - 2: overweight level 1
    - 3: overweight level 2
    - 4: obesity type 1
    - 5: obesity type 2
    - 6: obesity type 3
- The rest of the features are already numeric and did not need to be transformed:
    - NCP: How many meals do you eat daily?
    - CH20: How much water do you drink daily?
    - FAF: How often do you have physical activity?
    - TUE: How much time do you use technological devices such as cell phone, video games, television, computers, and others?

Cross Validation

All models were evaluated using 10-fold stratified cross-validation. K-fold cross-validation is used in this paper to ensure robust model evaluation. It improves model generalization by dividing the dataset into k subsets, ensuring every data point is used for both training and testing, which reduces variance and prevents overfitting. Stratified sampling is applied to maintain the same class distribution across all folds.

Techniques

We use 3 major techniques in order to predict the obesity level. They will be discussed in their own separate sections:

- Linear Regression
- SVM
- Multi-Layer Perceptron

Linear Regression

For this method, we performed linear regression on the data, but we preprocessed the data using polynomial features. We generated a new feature matrix consisting of all polynomial combinations of the features with degree less than or equal to the specified degree. We processed the data for degrees 1, 2, and 3. We used the sklearn.preprocessing.PolynomialFeatures library.

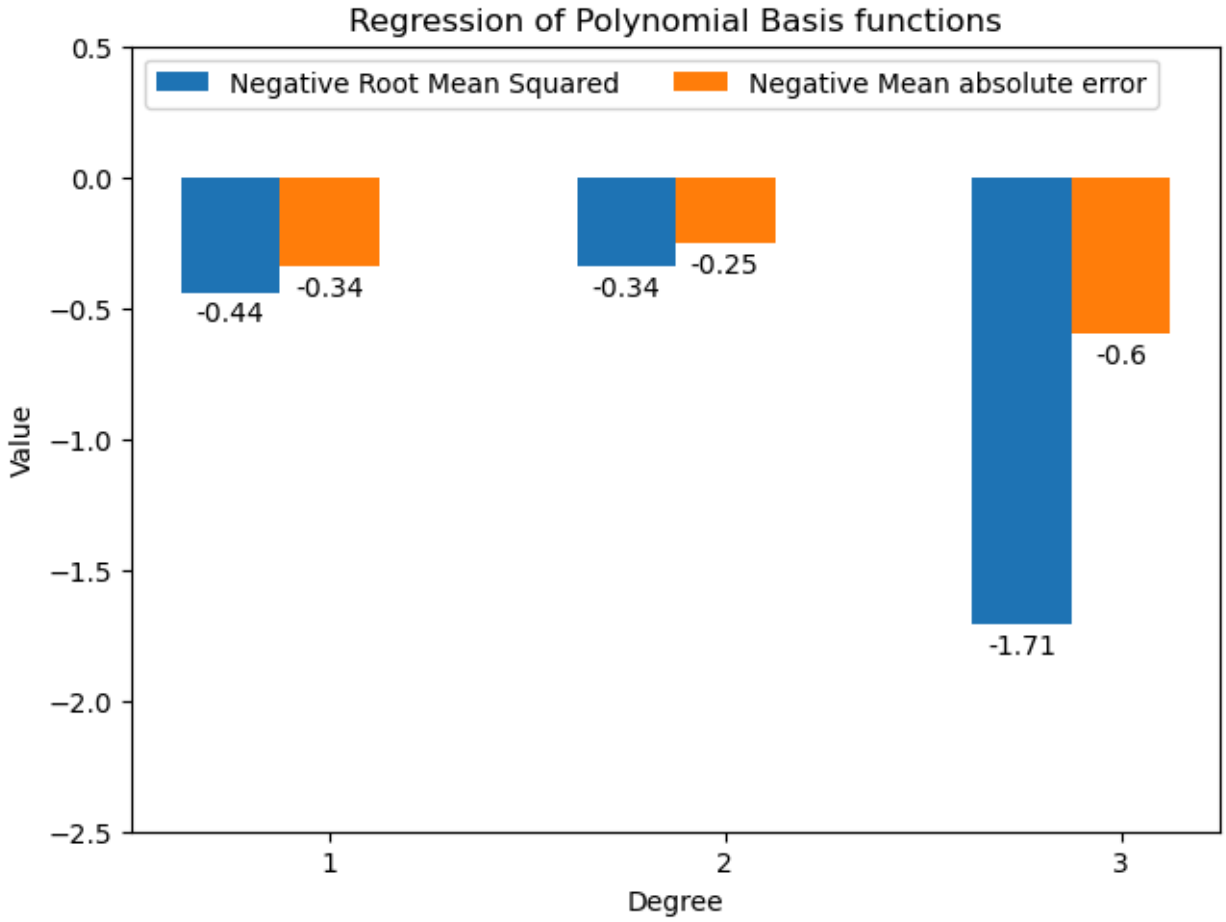
We used 2 scoring systems

- Mean Absolute Error(MAE)
- Root Mean Square(RMS)

Out of the 16 features, we wanted to see which features contributed or affected the 2 scoring systems listed above. In order to do this, we would run the regression several times and remove a different feature and see how the scoring system affected it.

Results

These are the results of our regression basis functions



For the negative root mean squared and negative mean absolute error, a higher score is considered better since we want to minimize these values. As you can see, polynomial with degree 2 worked the best. We believe that polynomial with level 3 starting performing worse because of overfitting.

Parameters removed	Root Mean Squared	Mean Absolute Error
No Params removed	0.35	0.25

Gender	0.35	0.25
Age	0.35	0.25
Height	0.53	0.38
Weight	1.13	0.83
family_history	0.34	0.25
FAVC - High calorie food consumption	0.34	0.25
FCVC - normally eat vegetables	0.35	0.25
NCP - # main meals a day	0.35	0.25
CAEC - food between meals	0.34	0.25
SMOKE	0.33	0.24
CH2O - water consumed	0.35	0.25
SCC - monitor calories	0.33	0.24
FAF - regular physical activity	0.34	0.25
TUE - time on devices	0.33	0.24
CALC - alcohol frequency	0.33	0.24
MTRANS - type of transportation used	0.34	0.25

If we use all the features to build our linear regression model, the RMS value is 0.35 and MAE value is 0.25. If we remove any of the features from the model except weight or height, we find that the RMS value and MAE values stay relatively the same. When we removed height, the RMS/MAE values increased to 0.53/0.38 and when we removed weight, the RMS/MAE values skyrocketed to 1.13/0.83. The feature that affected the model the most was weight which made sense.

We then tried to build a model with only height and weight, the 2 parameters that we saw contributed the most to the model and these were our results

All except Height/Weight	0.37	0.29

Building a model with only these two parameters gave us a model that was worse than including all 16 parameters. This shows that even though the contribution to the model for the other 14

parameters was small, all of them combined did improve the quality of the model and should still be used.

SVM

For this method, we use every possible combinations of the following parameters:

- Gamma: this is a hyperparameter that controls the influence of each training sample, especially in non-linear kernels. We use two values: “auto” and “scale.”
- Kernels:
    - Linear kernel: simple dot product between two samples
    - Polynomial kernels: Kernel in the form of $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\mathsf{T}\mathbf{y} + c)^d$ where d is the degree of the polynomial. We use degrees 2 and 3.
    - Radial Basis Function kernel (RBF): $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$
    - Sigmoid kernel: $K(X,Y)$=tanh(γ$\cdot XTY+r$)
- Regularization coefficient: we use C values from 0.01, 0.1, 1, 10, 100

And each method is scored by 3 scoring systems:

- Accuracy
- Mean Absolute Error (MAE)
- Root Mean Square (RMS)

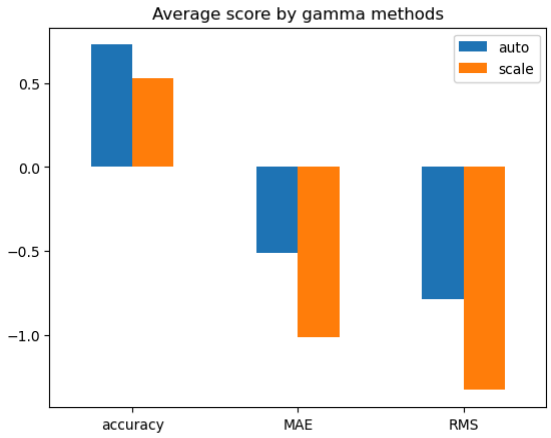
MAE and RMS are similar to the regression method. But this time, because SVM is a classifier, we also could use the accuracy method, that is, how well the model predicts the actual label as opposed to the other labels. The distance between actual label and predicted label is ignored.

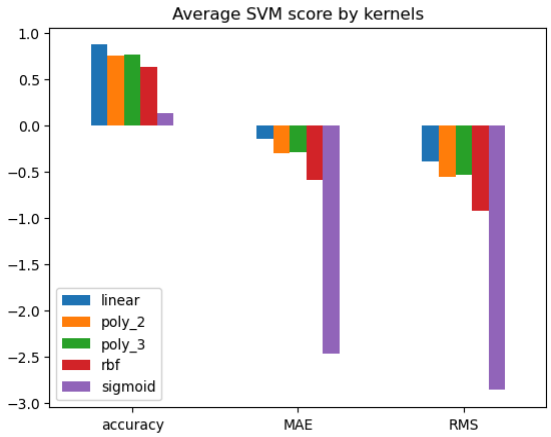
Every model’s performance and scores are saved in a multi-index dataframe (see notebook), and the results are analyzed

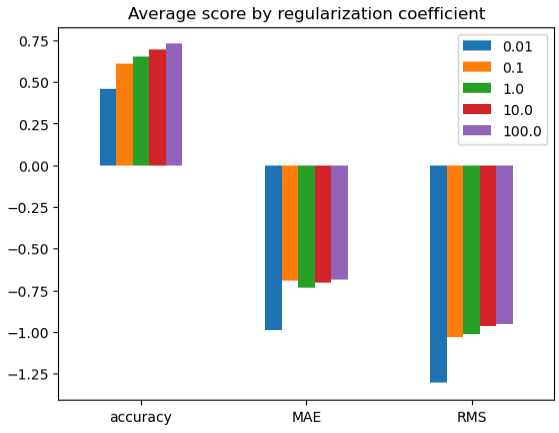
Results

Average scores by gamma, kernel, regularization coefficients

We plot the average results organized by gamma, kernel, and regularization coefficients







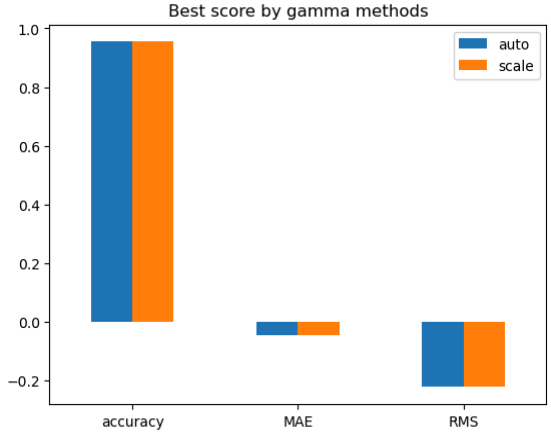
The actual values behind the plots are also printed in the notebook output cells. The higher the bar, the better model performance is. Especially with MAE/RMS, similar to the regression method, the loss function is negative MAE / negative RMS. So the score values are negative, but the higher the number (the less negative), the better.

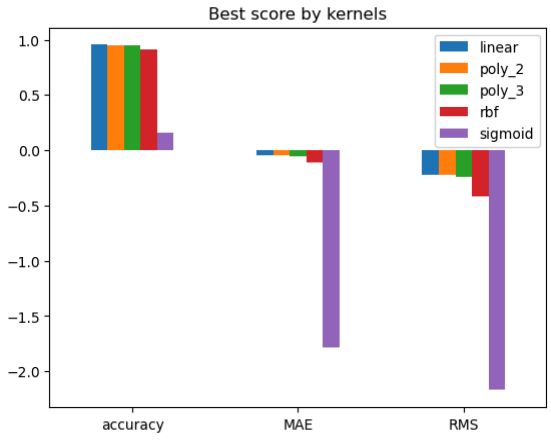
Observations from these plots:

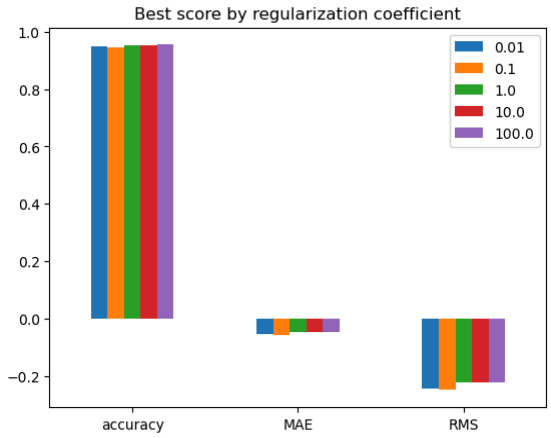
- Gamma='auto' performs better than gamma='scale' in all categories. This is because we let SVC choose the correct values of gamma as determined through cross-validation process. Gamma='scale' scales the gamma according to the magnitude of each feature.
- Linear kernel performs the best. The polynomial and RBF kernels perform decently well, and the sigmoid kernel performs the worst. A lot of this has to do with the geometry of the data, and as SVM is fundamentally a geometric classifier, the performance of each kernel is very sensitive to the geometry of the datasets. Later when we show the plots, one can visually see how the linear kernel performs the best.
- The higher regularization coefficient is, the better the model performs. This is not surprising as regularization is designed to prevent overfitting. The higher C value tends to choose smaller-margin in order to classify more points, whereas the lower C value tends to choose larger margin even if it misclassifies a few points. Our observation is also that the higher C value takes longer to train, most likely because the optimization problem is more difficult to solve.

Maximum scores by gamma, kernel, regularization coefficients

We also pick the best-performing model in each scoring category, categorized by gamma, kernel, and regularization coefficients.







The actual values behind the plots are also printed in the notebook output cells.
A few observations:

- There is no discernible difference between gamma='auto' and gamma='scale'. In other words, the best performing gamma='auto' model chooses the right gamma value and it's most likely the gamma that is scaled due to variations in feature values. The lesson here is to let SVC choose gamma='auto' as it will pick the best value as it goes through the cross-validation process.
- Even though polynomial degree 3 outperformed polynomial degree 2 on average, the best of polynomial degree 2 outperformed degree 3.
- Regularization still does improve the performance but not as much. One way to read this is: if a model already does well by virtue of having the right gamma and kernel, more regularization won't help that much because the model already classifies most of the points correctly. But if a model doesn't do well (e.g. because of a bad kernel choice), regularization helps a lot by encouraging it to prioritize correctness over margin width.

Best SVM Model

The best SVM model is printed by this command: auto gamma, linear kernel, with regularization coefficient of 100

```
print("Maximum score for each scoring function")
print(df_multiIndex.groupby(level=[3]).idxmax())
print(df_multiIndex.groupby(level=[3]).max())
```

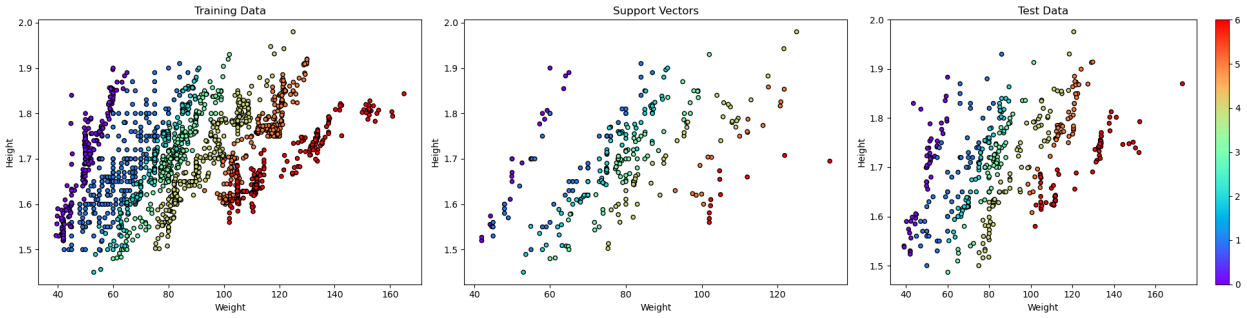
✓ [11] 33ms

```
Maximum score for each scoring function
                                                                    value
accuracy                                        (scale, linear, 100.0, accuracy)
neg_mean_absolute_error           (scale, linear, 100.0, neg_mean_absolute_error)
neg_root_mean_squared_error  (scale, linear, 100.0, neg_root_mean_squared_e...
                                    value
accuracy                         0.956770
neg_mean_absolute_error         -0.045006
neg_root_mean_squared_error     -0.213660
```

Support Vector Plots

Upon training and testing, we also have access to the support vectors that the training process produced. These are training samples that lie on the maximum-margin hyperplanes. In the notebook, we produced a plot of every combination of two features, to see the training samples, support vectors, and testing samples. Some notable ones we will discuss here are:

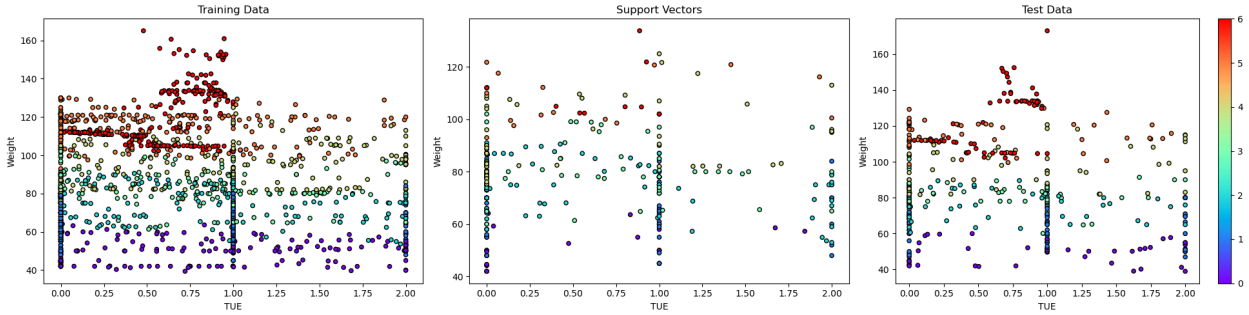
Height vs Weight



In the plot above, the x-axis is weight, the y-axis is height, and the color-coding is the obesity level, with purple being underweight, blue being normal, and red being the most overweight. Unsurprisingly, in the training data, the relationship between height and weight is increasing and looks “approximately linear.” This visual inspection is confirmed by the support vectors that almost perfectly fall in a straight line for each color category.

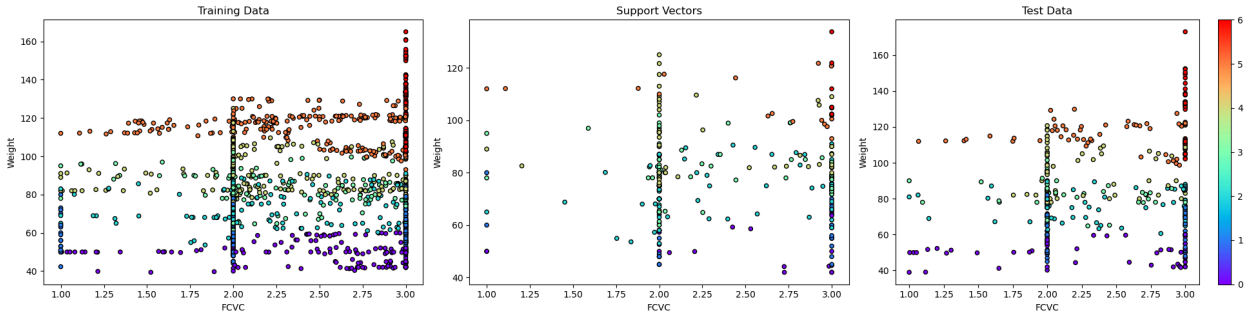
The fact that the support vectors do not fall perfectly in a straight line means that the other features also contributed to the final hyperplane. In fact, as the regression method reveals, all features contributed positively to the classification / regression results. If we take any single feature out, the performance suffers slightly. However, height and weight remain the two most significant predictors of the obesity level.

Weight vs TUE



In the plot above, the x-axis is TUE (technological usage such as cell phone, video games, etc) and the y-axis is weight. Unlike height vs. weight, there is no obvious linear relationship between the two variables, although we could see that for each TUE value, the obesity level increases with weight, but the thresholds are different for each TUE value (as shown by the support vector plot in the middle). This is one case where we think TUE is still valuable as a predictor but not by a lot.

Weight vs FCVC



In the plot above, the x-axis is FCVC (Fiber and vegetable consumption daily) and the y-axis is weight. Similar to weight vs. TUE, there is no obvious linear relationship, and in fact the color gradient looks almost horizontal. However, the support vectors are much sparser for the low values of FCVC and are far more densely populated for the higher values of FCVC. However, for the low values of FCVC, the color-coding bands are perfectly horizontal, both in the training data and in the support vector plots.

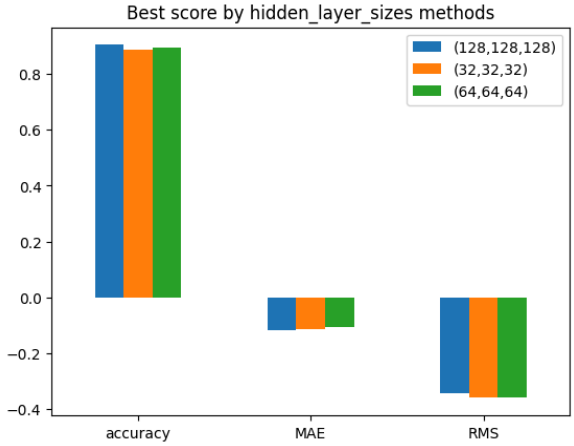
One was to understand this graph is, the value of FCVC as a predictor is by imposing a threshold. If an individual eats enough vegetables, then the other factors can come into play. But if he/she does not eat enough vegetables, then weight alone is almost sufficient to predict the obesity level.

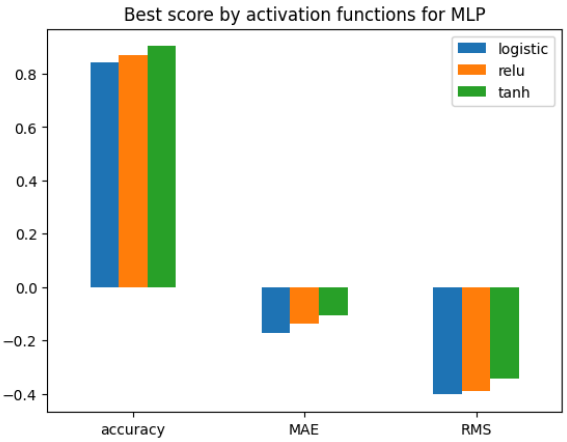
Multi-Layer Perceptron

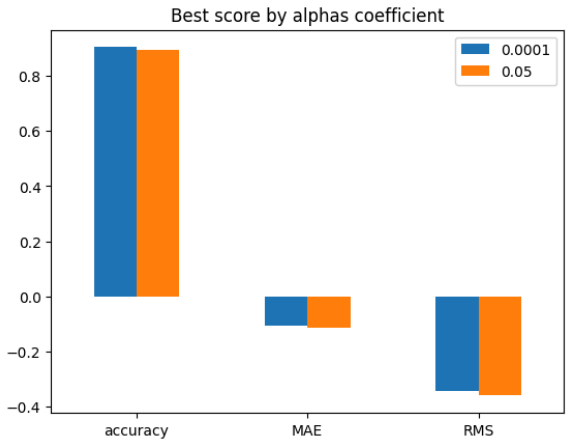
For the Multi-Layer Perceptron, we use different combinations of the following parameters:

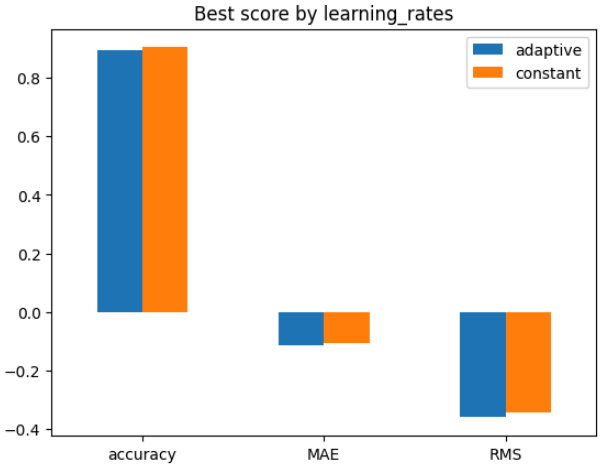
- Hidden Layer Sizes: this is a hyperparameter that controls how large and deep our neural network is by deciding the number of nodes and layers. We tested size of (32,32,32), (64,64,64), and (128,128,128).
- Activations:
    - tanh: This function maps the input to a range between -1 and 1.
    - relu: This function outputs zero for negative values and the input value itself for positive values.
    - logistic: The logistic function, or sigmoid, maps the input to a range between 0 and 1.
- Alphas: we use the values of 0.0001 and 0.05
- Learning Rates: we tested the constant and adaptive learning rates.

Results









The best performing MLP mode was able to achieve an accuracy of 91% with the hyperparameters of (128,128,128) for the hidden layer size, tanh for the activation function, 0.0001 for the alpha value, and a constant learning rate. In the charts above, we see that the activation function had the biggest difference in performance for the MLP model. Similar to the SVM model, we see logistic, or sigmoid, as the worst performing activation function while tanh outperformed the other activation functions. Values like the alphas and learning rates had smaller impacts on the overall performance.

Cross-method Data Comparison

A few observations when comparing results from several methods:

- In the regression analysis, we found that cubic basis function does not do well at all, whereas quadratic basis function performs the best. We hypothesized that the cubic basis function overfits the data
    - However, upon running SVM with linear, quadratic, and cubic kernels, we found that quadratic and cubic functions behave about the same, and they're worse than linear kernel. The best of the quadratic kernel did outperform the best of the cubic kernel (this is with high regularization coefficient and gamma=auto). It does support the hypothesis that quadratic kernels can fit the shape of the data better, but does not explain why the linear kernel performs the best whereas in the regression case, linear basis functions perform slightly worse than quadratic basis.
- In the SVM analysis, we found that the sigmoid kernel performs the worst. This observation is confirmed by the MLP. When we use sigmoid (logistic) activation function, the performance is the worst, although it's not that far behind tanh and relu. One way to explain this is, in the MLP case, the sigmoid is only applied to the neurons that have been activated in the previous layer, so not all "corners" of the data are getting sigmoid function. And the activation itself is computed via a linear function, which SVM already showed as the best performing kernel, which means the geometry of the data is somewhat friendly to linear basis and kernels.

Best-performing model

After inspecting the results of all methods, the best performing model is SVM with auto gamma, linear kernel, and regularization coefficient of 100. This can be explained by the following observations:

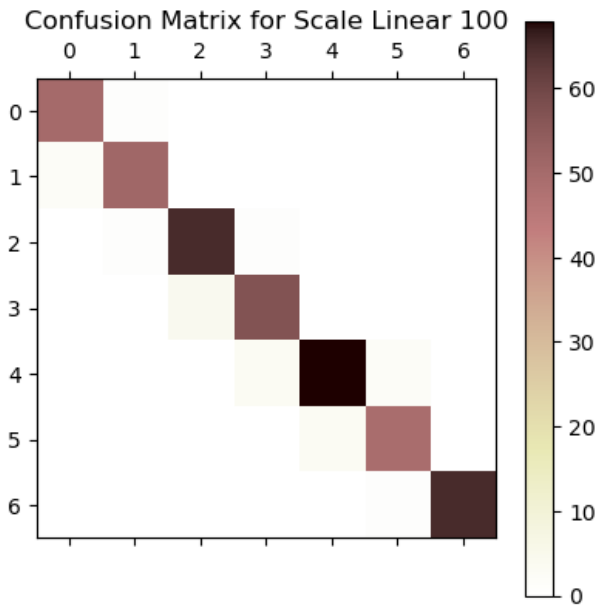
- The data geometry is primarily linear, and most of the labels can be explained by a linear relationship between height and weight
- Regularization coefficient optimizes for correctly classifying the points at the expense of margin width

The confusion matrix and the precision/recall table of this model on the testing data is displayed below. The heatmap version of the confusion matrix that we printed for the poster session is also displayed below. For this model, we were able to get 95.7% accuracy to predict the actual obesity class, and all of our misclassifications are within 1 label away from the true label.

```
SVC(C=100.0, degree=0, kernel='linear')
0.9574468085106383
0.20628424925175867
0.0425531914893617
[[50  1  0  0  0  0  0]
 [ 2 51  0  0  0  0  0]
 [ 0  1 65  1  0  0  0]
 [ 0  0  4 57  0  0  0]
 [ 0  0  0  3 68  2  0]
 [ 0  0  0  0  3 49  0]
 [ 0  0  0  0  0  1 65]]
              precision    recall  f1-score   support

           0       0.96      0.98      0.97        51
           1       0.96      0.96      0.96        53
           2       0.94      0.97      0.96        67
           3       0.93      0.93      0.93        61
           4       0.96      0.93      0.94        73
           5       0.94      0.94      0.94        52
           6       1.00      0.98      0.99        66

    accuracy                           0.96       423
   macro avg       0.96      0.96      0.96       423
weighted avg       0.96      0.96      0.96       423
```



Future Work

If we had more time for this project we would like to explore the following (or we can do this in the future if we have a chance to extend this project):

- Explore more permutations of features to see which one performs the best without overfitting
- Explore more classes of classifiers / regressors beyond linear regression, SVM, and MLP
- Apply more meta-techniques such as adaptive boosting

Conclusion

Starting with the obesity datasets from 2111 individuals in Mexico, Peru, and Colombia, we applied regression, SVM, and MLP techniques in order to predict the obesity levels of the individuals. By using ~1600 samples as training data and ~400 samples as testing data, we were able to achieve 95.7% accuracy in predicting the correct obesity label. Furthermore, the misclassification in the testing sets were all within 1 obesity level away from the true label.