Department of Biomedical Engineering

Senor Systems and Data Analytics (BMEN 415) Winter 2022

Final Project Report Group 12

Submitted: April 12th, 2022

Team Members:

Carter McIlwraith	30091230
Davina Dang	30087063
Mitchell Rhead	10185289
Sadia Khan	30090271
Marc Vincent Raranggor	30090829

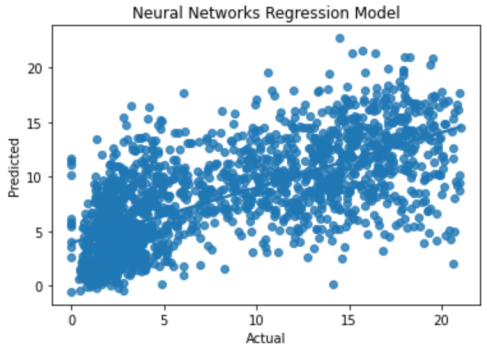
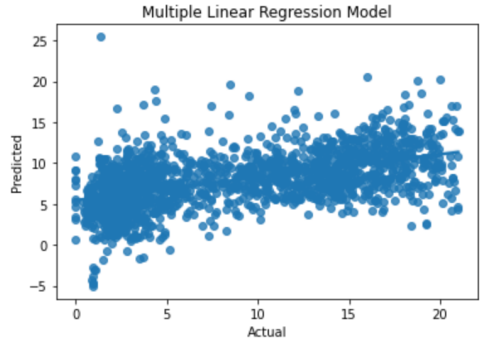
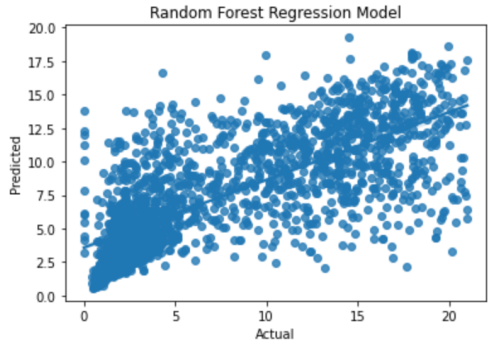
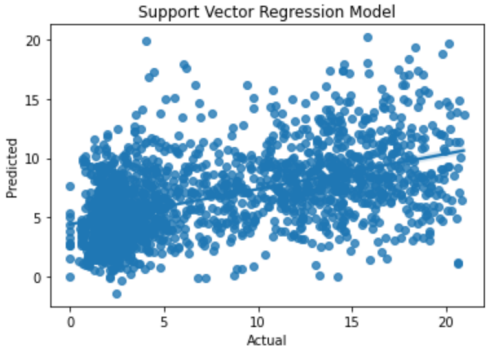
Introduction

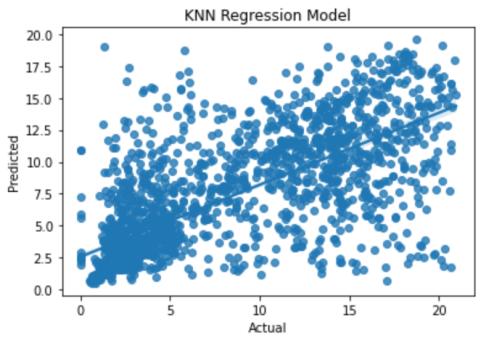
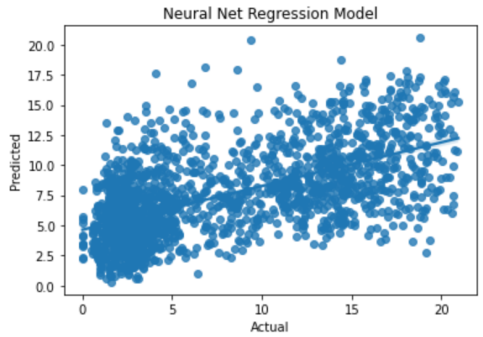
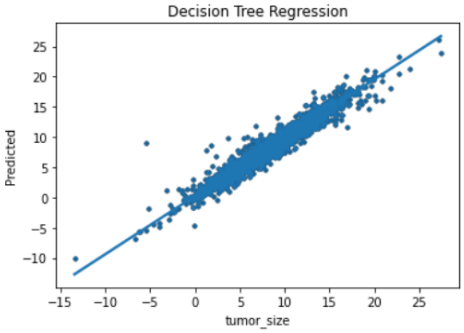
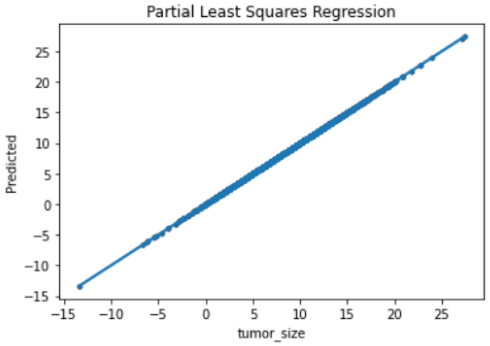
For the regression dataset [1], we are using melanoma tumour size prediction. Melanoma is a type of skin cancer that occurs within the pigment-making cells called melanocytes. Melanoma tumours can be found in various different sizes, shapes, and colours. Using the mass of the area, size of the area, the ratio of normal malign, the size of the damaged area, total exposed area, the standard deviation of malign skin, and the error in malign skin measurements, an accurate prediction of tumour size can be determined. The target of this dataset is to predict the size of a melanoma tumour using the attributes above.

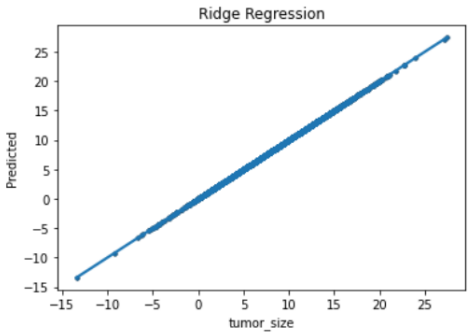
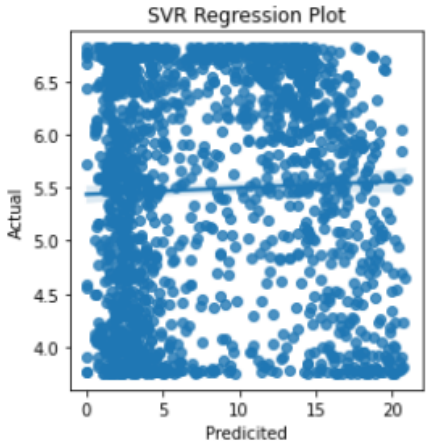
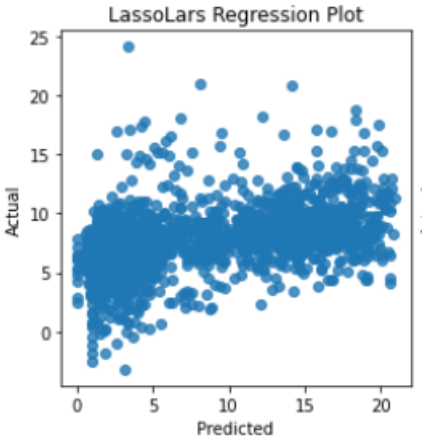
For the classification dataset [2], we are using the data from a study that predicts diabetes occurrence. Features of the data include the number of times one was pregnant, plasma glucose concentration, diastolic blood pressure (mm Hg), triceps skinfold thickness (mm), 2-hour serum insulin (μU/ml), body mass index (kg/m^2), diabetes pedigree function and age (years). The target of this data is to classify if an individual has diabetes or not.

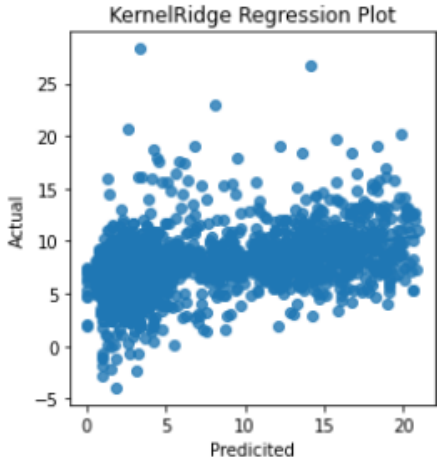
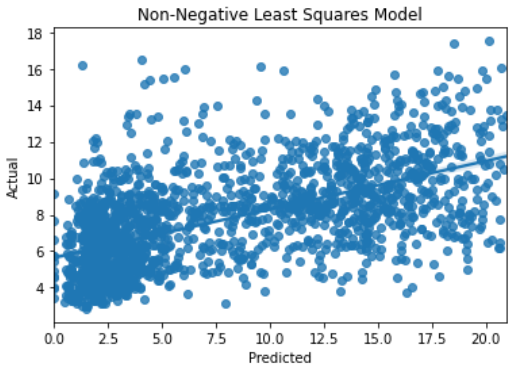
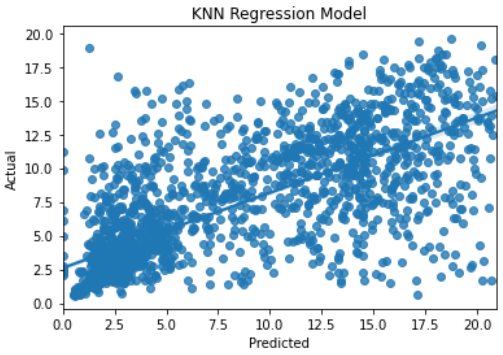
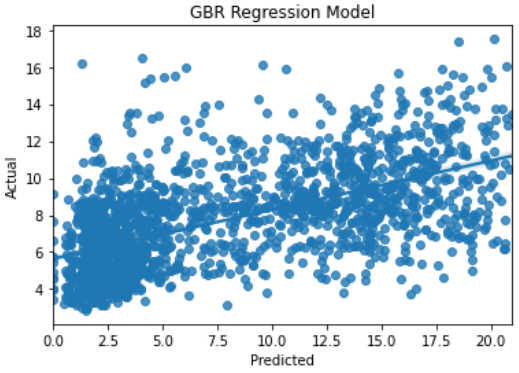
For the image classification dataset [3], we are using a dataset of various ocular diseases. This dataset is meant to represent a “real-world” application of image classification. The attributes of the dataset consist of Normal (N), Diabetes (D), Glaucoma (G), Cataract (C), Age-related Macular Degeneration (A), Hypertension (H), Pathological Myopia (M), and Other (O). The target of the dataset is to accurately diagnose the patients' ocular disease based. To classify the images, we decided to use TensorFlow and develop our own Convolution Neural Network model (CNN). CNNs have a wide range of applications, however, are mostly used for image analysis as they have the ability to develop a representation of the 2D images being studied. This feature allows for CNNs to identify the position and the scale in various different structures in the images. This is very useful for our purposes as the identifiers for the different ocular diseases are in various different positions and scales. This makes CNN an optimal choice for this task. In theory, using a convolutional neural network model should allow us to be able to accurately predict and identify the different ocular diseases within the data set.

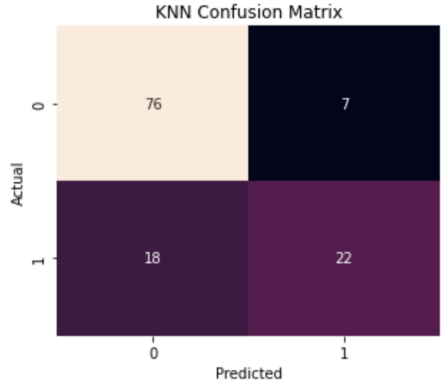
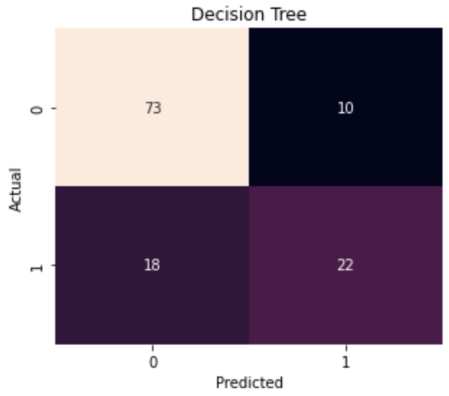
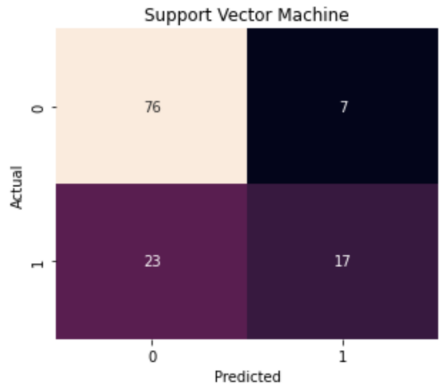
Evaluation of Models

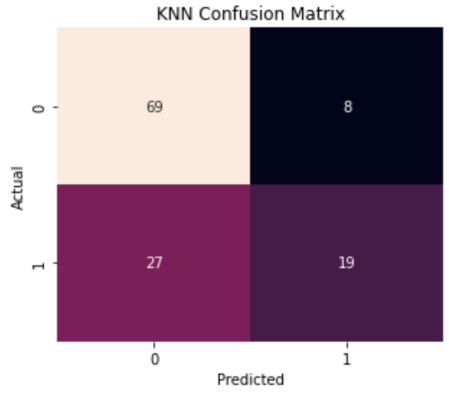
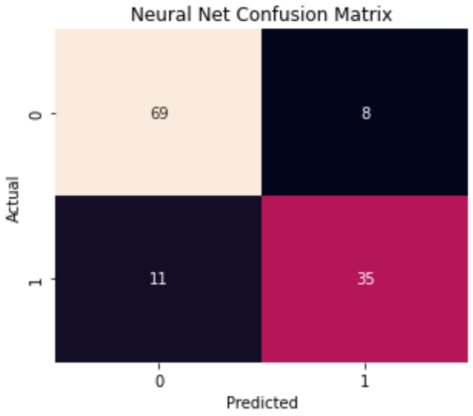
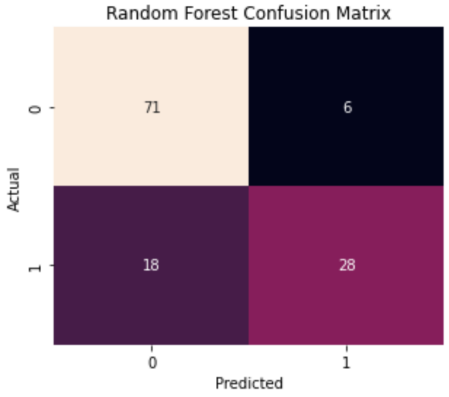
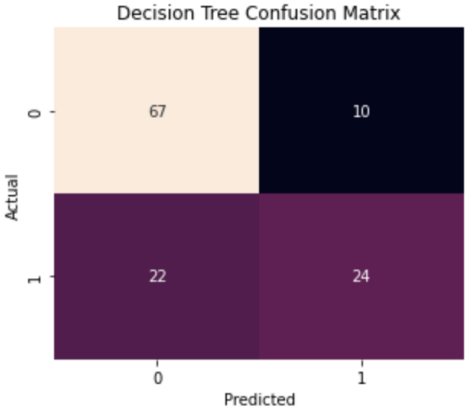
Regression Model Comparison				
Group Member	**Model**	**Regplot**	**Mean Squared Error**	**R^2 Score**
Marc	Neural nets		20.60	0.47
	Multiple linear regression		27.60	0.28
	Random forest		17.85	0.53
Sadia	Support vector machines (SVM)		27.57	0.23

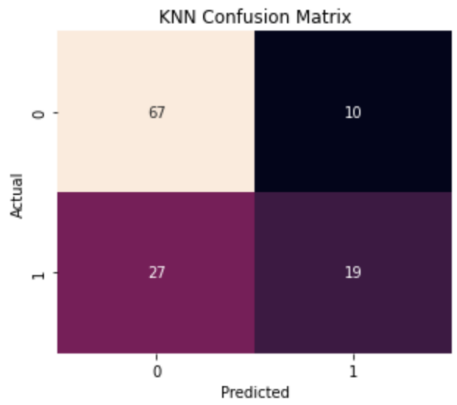
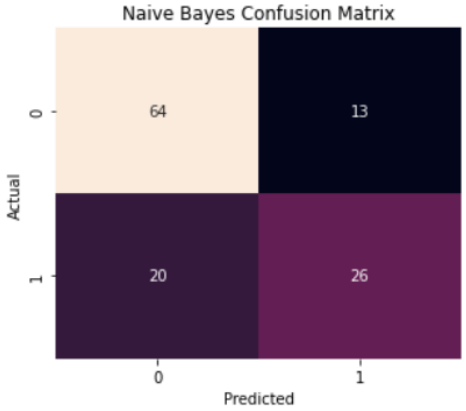
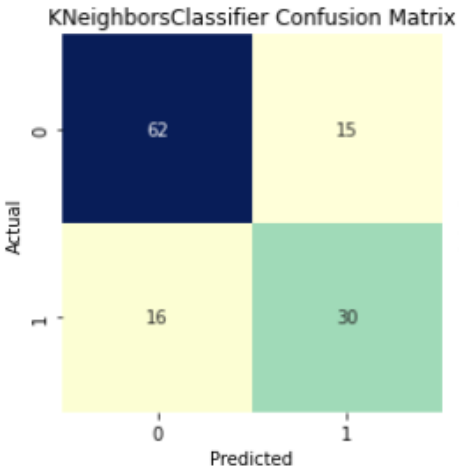
	KNN with 4 neighbours		18.51	0.49
	Neural nets		22.84	0.36
Mitchell	Decision tree		0.4786	0.9543
	Partial least square (PLS)		0.0000	1.0000

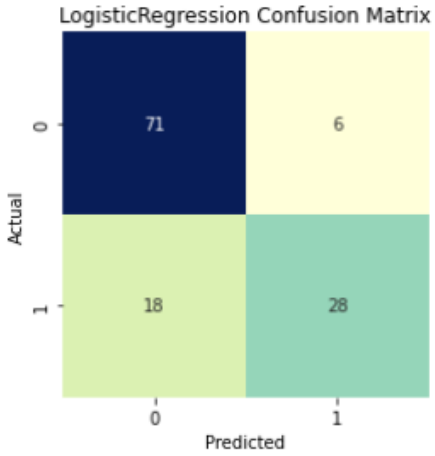
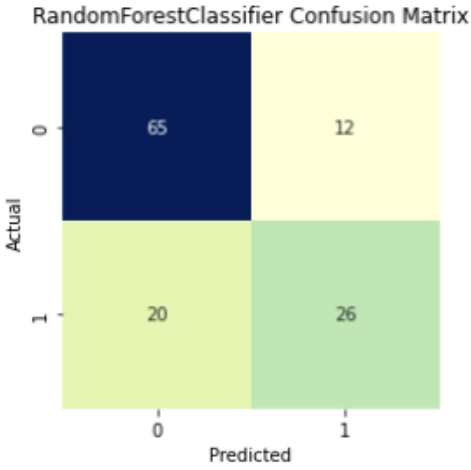
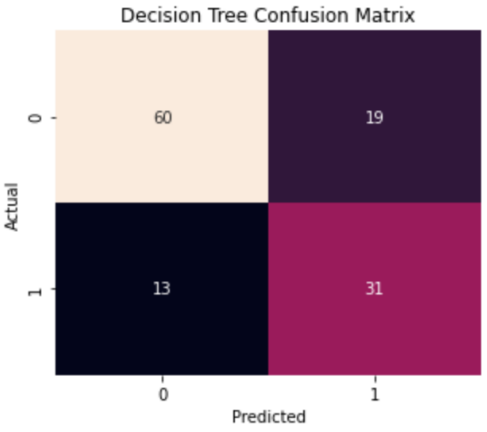
	Ridge		0.00000012	0.999999 99
Carter	Support Vector Regression (SVR)		41.3	-0.15
	LARS Lasso		29.18	0.19

	Kernal ridge		30.13	0.17
Davina	Non-Negative Least Squares		29.18	0.19
	K-Nearest Neighbours		17.93	0.50
	Gradient Boosting		24.18	0.33

Classification Model Comparison			
Group Member	**Model**	**Confusion Matrix**	**Accuracy Score (%)**
Marc	KNN with 11 neighbours		79.67
	Decision tree		77.24
	Support vector machine (SVM)		75.61

Sadia	KNN with 4 neighbours		71.54
	Neural nets		84.55
	Random forest		80.49
Mitch	Decision Tree		73.98

	KNN with 9 neighbours		69.92
	Naive Bayes		73.17
Carter	KNN with 3 neighbours		74.80

	Logistic regression		80.49
	Random forest		74.80
Davina	Decision Tree		73.98

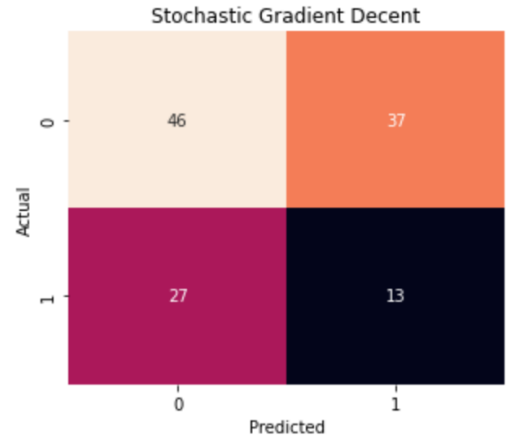
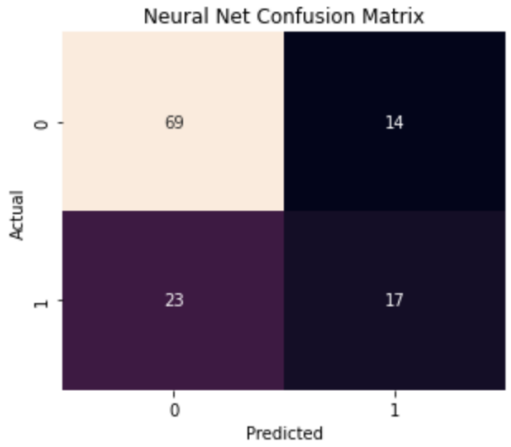
	Stochastic Gradient Descent		71.54
	Neural Networks		69.92

Image Classification Model Evaluation

	precision	recall	f1-score	support
N	0.641361	0.606436	0.623410	404.0
D	0.558394	0.733813	0.634197	417.0
G	0.866667	0.188406	0.309524	69.0
C	0.854545	0.635135	0.728682	74.0
A	0.428571	0.039474	0.072289	76.0
H	0.500000	0.151515	0.232558	33.0
M	0.629630	0.326923	0.430380	52.0
O	0.500000	0.111940	0.182927	268.0
micro avg	0.603261	0.478105	0.533440	1393.0
macro avg	0.622396	0.349205	0.401746	1393.0
weighted avg	0.596417	0.478105	0.485405	1393.0
samples avg	0.539127	0.502883	0.512768	1393.0

Figure 1: Precision, Recall, f1-Score, and Support Values of the Model for each Image Category

Report Summary

To compare our **regression** models, our group evaluated the mean squared error (MSE) and R^2 score. The most accurate model was the Partial Least Squares (PLS) model, with perfect accuracy and zero error. This model did not have any manipulation performed to the original dataset, so the model was trained/tested with all the data; some values in the output *tumor_size* column were negative, and so the model was built to allow negative outputs. Otherwise, PLS has the advantage of being able to consider the variability of dependent variables, perhaps contributing to its effectiveness. The least accurate model was Support Vector Regression (SVR) with an R^2 score of -0.15, making it less accurate than the mean of the data. MSE was 41.3, which was also extremely poor. The low performance of SVR could be attributed to the noise present in the data set. For datasets containing lots of noise or overlapping target classes SVR is known to underperform. Additionally, SVR is not very accurate for large datasets as overfitting becomes a problem. These can lead to the model not accurately predicting the target.

In order to directly compare the individual **classification** models, each group member calculated for the accuracy score. The accuracy is a simple ratio of the correctly predicted values to the total observations (predictions). In addition to the accuracy score, we each determined the confusion matrix of each classification, an effective way to depict the predicted results and class distribution in the data, along with the breakdown of error types. These calculations allow us to directly compare the efficiency of each of our models numerically. The classification model that performed the best is the neural networks due to the fact that it is an efficient model for non-linear data with a large number of inputs, providing an accuracy score of 84.55% . Whereas the classification model that was observed to perform the worst was K-nearest neighbours with nine neighbours, giving us an accuracy

score of 69.92%. This is due to the fact that KNN does not work well with high dimensionality, as it complicates the calculating process for distance between each dimension, in addition to the required scaling in each dimension.

The classification model used in the image input problem was classification through convolutional neural networks (CNN). The model runs very smoothly, however, the time for it to run all the way through could vary from 10 minutes to 13 hours. The cause of this running time difference is the number of epochs inputted to train. Each epoch runs five to 10 minutes, thus, if one inputs 100 epochs to train, the model could run for more than 12 hours. From our results, we determined that training 83 epochs would give the best results due to it having an accuracy of 86.1% and only having a loss of 10.7%. However, due to time constraints, the epochs inputted to train were 100 epochs. As was learned in class, an excessive amount of epochs could also decrease the accuracy of the model. In our case, when testing the model, the accuracy received was 55.9% with a loss value of 35.8%. This low value of accuracy could be traced back to the number of epochs selected to be trained. Figure 1 displays the precision percentage, recall percentage, f-1 scores, and support values for each category of the images.

In comparison to other models like Dense Neural Network (DNN), CNN would still be a better choice for image classification. If it took around 80 epochs for the model with more than 12,000 input images to converse using CNN, it would take more than twice the number of epochs to be trained if DNN was used. CNN uses a method called convolution, hence the name, which applies a filter on the image input to filter information to create a feature map. This feature map is on a layer called the convolutional layer which autonomously recognizes the features of the image. The convolutional layer increases the accuracy of the model recognition because, unlike DNN, the neurons are not densely connected to every neuron in the next layer [4].

Group Report Contributions:

- Carter: Built 3 regression models (SVR, LARS Lasso, Kernal ridge), 3 classification models (KNN with 3 neighbours, Logistic regression, Random Forest), built the CNN for image classification, wrote the chosen hypothesis for the image classification case study
- Davina: Built 3 classification models (Decision tree, Stochastic Gradient Descent, Neural networks), 3 regression models (Non-negative least squares, K-nearest neighbours, Gradient boosting descent) Wrote the comparison of the classification models.
- Sadia: Built 3 regression models (SVM, KNN with 4 neighbours, Neural Nets), 3 classification models (KNN with 4 neighbours, Neural Nets, Random Forest), wrote and modified introduction, the evaluation model section for both the regression and classification model comparison, created the GitHub and organised the folders
- Marc: Built 3 regression models (Neural Networks, MLR, Random Forest), 3 classification models (kNN with 11 neighbours, Decision Tree, SVM), built the CNN for image input classification and wrote the summary section for the results of the image input problem model.
- Mitchell: Built 3 regression models (Decision tree, PLS, Ridge), 3 regression models (Decision Tree, KNN, Naive Bayes), writing regression summary and drawing conclusions.

Git Repository

https://github.com/sadiatasneemkhan/BMEN_415_Project

References

1. A. Kumar, *"Machine Hack: Melanoma Tumor Size Prediction."* Kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/anmolkumar/machine-hack-melanoma-tumor-size-prediction?select=sample_submission.csv. [Accessed: 11-Apr-2022].

2. *"Diabetes Classification."* Kaggle.com. [Online]. Available: https://kaggle.com/competitions/diabetes-classification. [Accessed: 11-Apr-2022].

3. Larxel, *"Ocular Disease Recognition."* Kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/andrewmvd/ocular-disease-recognition-odir5k. [Accessed: 11-Apr-2022].

4. Y. Gavrilova, "What are convolutional neural networks?," *Serokell Software Development Company*, 03-Aug-2021. [Online]. Available: https://serokell.io/blog/introduction-to-convolutional-neural-networks. [Accessed: 12-Apr-2022].