Lab5 for STAT 415

Xiaolong Liu

May 2024

1 Introduction

In this lab, we will consider several methods for explaining the outputs of predictive models. We will focus on attribution methods, which try to weight the relative importance of inputs with respect to making a prediction. It is important to note there are many methods for considering the interactions between variables. This assignment will not consider those methods, because they get complicated quickly. Instead, we will focus on attribution methods such as LIME, Shapley Values, and SmoothGrad.

2 Tabular Data

We start with the Pima Indians Diabetes Database. This tabular database contains several health diagnostic measures, such as blood pressure, with the goal of predicting if a patient has diabetes.

There are some abnormal data points in this dataset:

1. **Glucose:** The Oral Glucose Tolerance Test (OGTT) is a diagnostic test for diabetes that measures how efficiently the body metabolizes glucose. A reading of 0 is not possible.
2. **Blood Pressure:** The minimum recorded value is 0, which is impossible for a living person.
3. **Skin Thickness:** The minimum recorded value is 0. It is impossible for a person to have a triceps skin fold thickness of zero.
4. **Insulin:** The minimum recorded value is 0. It is highly unusual for an individual, even one with diabetes who is insulin-dependent, to have a serum insulin level of zero unless there is a lab error or the individual has a rare form of diabetes.
5. **BMI:** The minimum recorded value is 0, indicating a weight of zero, which is impossible.

Therefore, we decide to use the mean of the non-zero values to impute these abnormal data.

2.1 The balance of the dataset with respect to the outcome variable

First, let's investigate the balance of the dataset with respect to the outcome variable. We find that the outcome variable indicating whether the patient is diabetic is undersampled. There are 34.9% of the total outcomes that have the patient as diabetic.

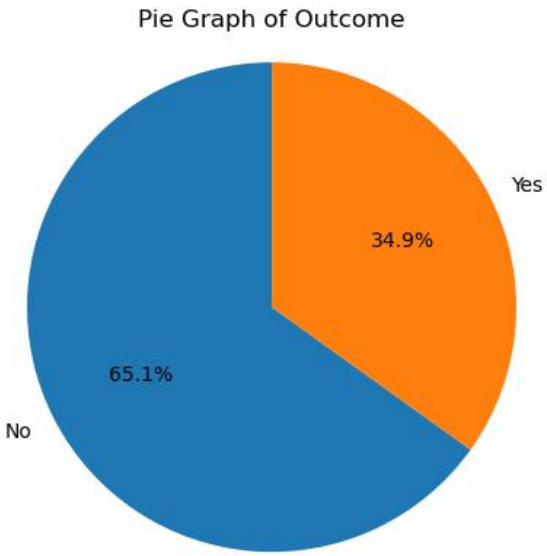

Figure 1: Initial distribution of outcome variable

To address this imbalance, we use the method **Oversampling the Minority Class** to make our dataset balanced.

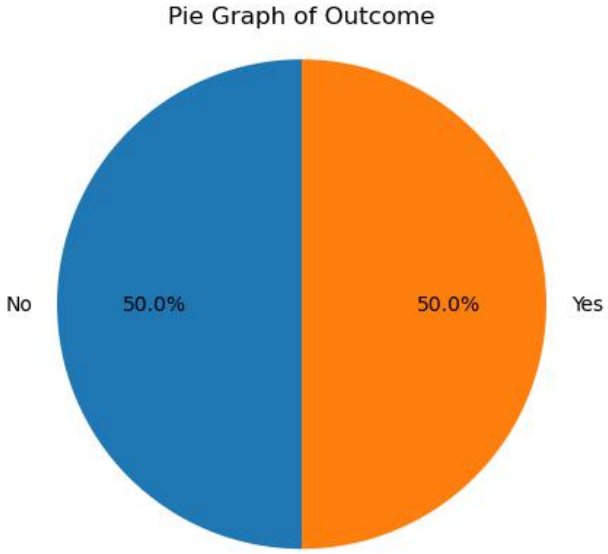

Figure 2: Distribution of outcome variable after applying oversampling

2.2 Logistic model with an L1 penalty

First, we fit a logistic regression model with L1 penalty on this dataset. We divide the dataset into three parts: train 60%, validation 20%, and test 20%. We employ grid search over the set $[0.001, 0.01, 0.1, 1, 10, 100]$ to find the optimal parameter for the penalty. The best parameter identified is 0.1 (note: the larger this value, the lesser the regularization effect).

```
Selected Features by Lasso:
         Feature  Coefficient
0    Pregnancies     0.351870
1        Glucose     1.039369
3  SkinThickness     0.180545
5            BMI     0.378512
7            Age     0.167457
```

The accuracy of our model is as follows:

- Train dataset: 74.5%
- Validation dataset: 74%
- Test dataset: 72.5%

The features selected by the model are **Pregnancies**, **Glucose**, **SkinThickness**, **BMI**, and **Age**.

2.3 Random forest

Next, we fit a Random Forest on the dataset. We used grid search to find the best parameters:

- `'bootstrap'`: False
- `'max_depth'`: 10
- `'max_features'`: 'sqrt'
- `'min_samples_leaf'`: 2
- `'min_samples_split'`: 2
- `'n_estimators'`: 150

The accuracy of this model is as follows:

- Train dataset: 97.2%
- Validation dataset: 81%
- Test dataset: 78.5%

Here's the feature importance of this model. Setting the threshold at 0.1, the selected features are **Glucose**, **Age**, and **BMI**.

```
Feature Importances:
 Glucose                    0.304321
Age                         0.163498
BMI                         0.162923
Insulin                     0.089511
DiabetesPedigreeFunction    0.082891
SkinThickness               0.075740
Pregnancies                 0.063526
BloodPressure               0.057589
```

Figure 3: Feature importance graph showing the most influential features.

2.4 LIME

We use LIME to explain the importance of each feature for the random forest model, particularly analyzing the 6th datapoint in the test dataset.

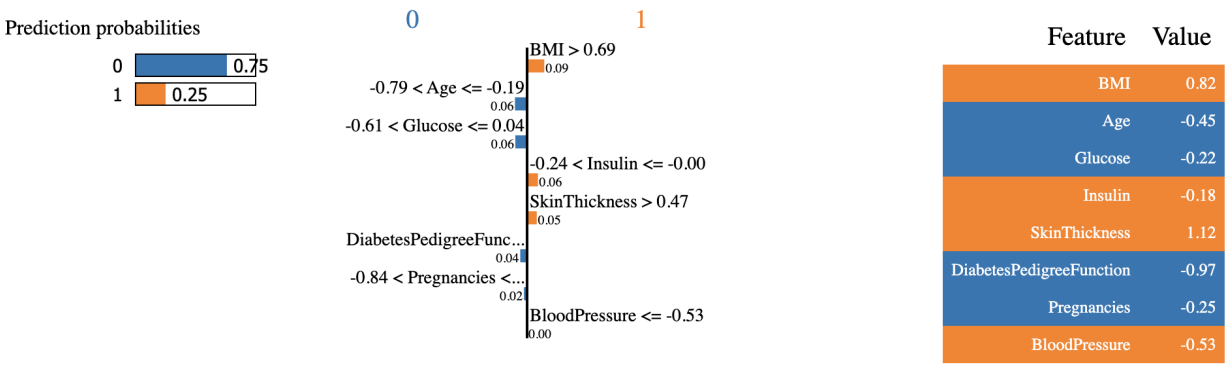

Figure 4: LIME interpretation for the 6th datapoint in the test dataset.

The model predicts with a probability of 75% that the instance belongs to class 0 and 25% to class 1. The bar chart on the right side of the output details the influence of each feature on the prediction for this particular instance. Features contributing positively towards class 1 are shown in orange, while those pushing the prediction towards class 0 are shown in blue. Each bar's length and direction indicate the weight and direction of the influence. For example, **BMI** (0.09): Positively influences the likelihood of being in class 1 when BMI is greater than 0.69. On the far right, the actual values of the features for the instance are listed. These values are what LIME used to compute how each feature influences the prediction. For instance, **BMI**: 0.82.

So, a **BMI** greater than 0.69 has the highest influence among all the features, followed by **Age** values between -0.79 and -0.19, and then **Glucose** values between -0.61 and 0.04.

2.5 Stability of LIME

Next, we apply LIME to different data points to evaluate the stability of the method. We observe that the influence of each feature varies across different data points, indicating some degree of instability. However, certain patterns remain consistent across the explanations: the features **Glucose**, **Age**, **BMI**, and **Insulin**

are almost always identified as important. In contrast, the feature **Blood Pressure** is almost always identified as the least important.

2nd data point:

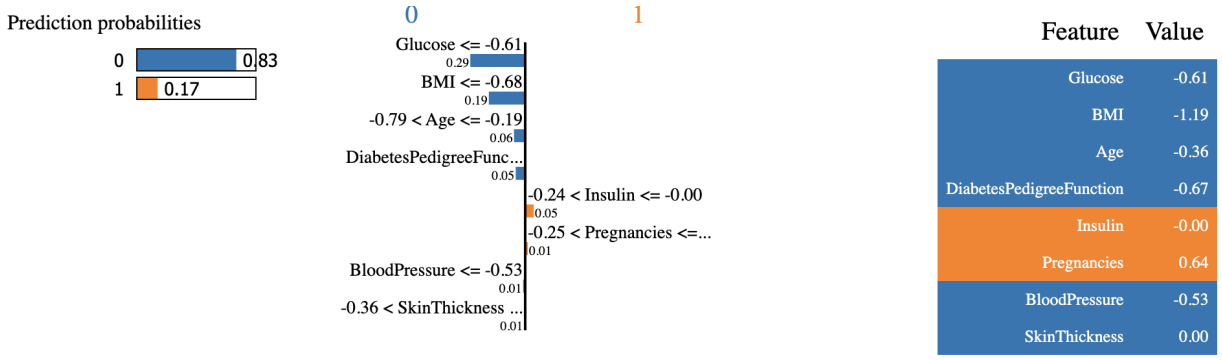

Figure 5: LIME explanation for the 2nd data point.

100th data point:

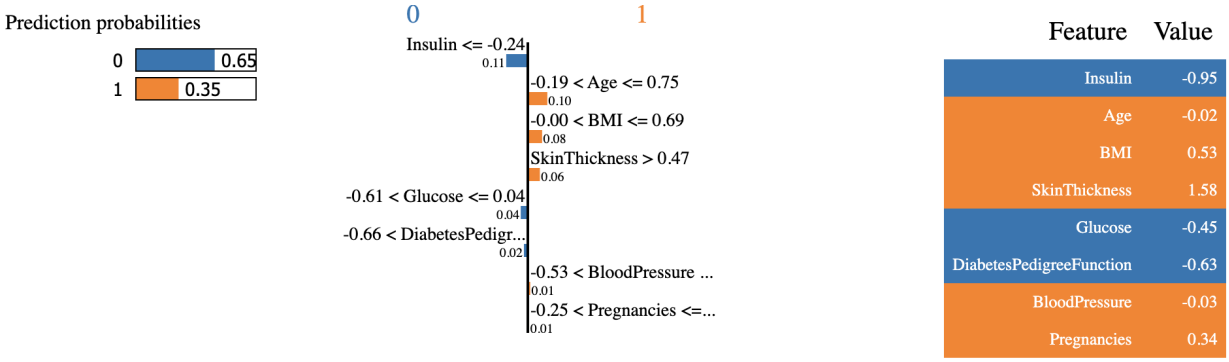

Figure 6: LIME explanation for the 100th data point.

61st data point:

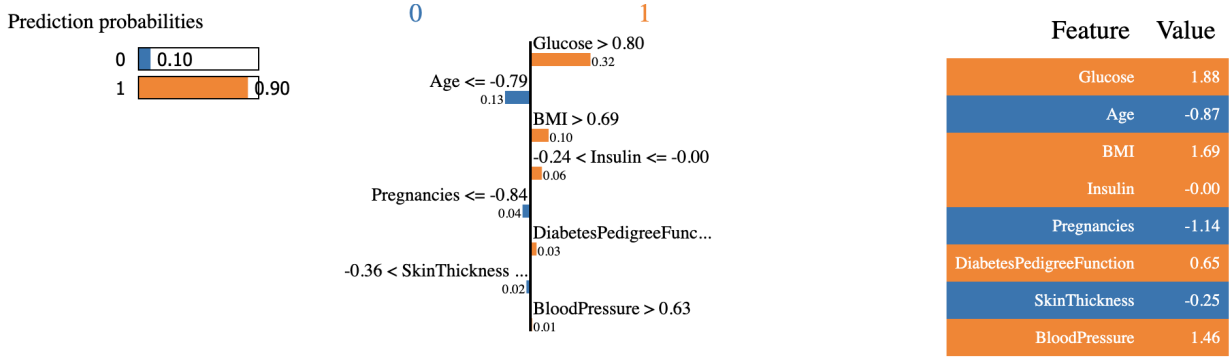

Figure 7: LIME explanation for the 61st data point.

2.6 Comparison for these models

Based on the previous results, we can draw the following conclusions:

For LIME applied to the 6th data point, the top five important features identified are **BMI**, **Age**, **Glucose**, **Insulin**, and **Skin Thickness**.

For the Random Forest importance analysis, the top five features are **Glucose**, **Age**, **BMI**, **Insulin**, and **Diabetes Pedigree Function**.

For the Logistic Regression model with an L1 penalty, the selected features are **Pregnancies**, **Glucose**, **Skin Thickness**, **BMI**, and **Age**.

All three methods consistently identify **Glucose**, **Age**, and **BMI** as important features. Additionally, both LIME and the Random Forest importance method select **Insulin**, whereas the Random Forest importance and the Logistic Regression with L1 penalty both highlight **Skin Thickness**.

However, the order of importance of these features varies among the methods.

3 Predictive Modeling on Animal Images

3.1 Load data and visualization

The ANIMAL-10N dataset is comprised of ten animal classes, with a total of $50,000$ training images and $5,000$ testing images. The objective is to develop a model that can accurately classify the type of animal depicted in any given image from this dataset.

First, we load the dataset. Each image within this dataset has dimensions of $64 \times 64 \times 3$. Below is the second image from the dataset:

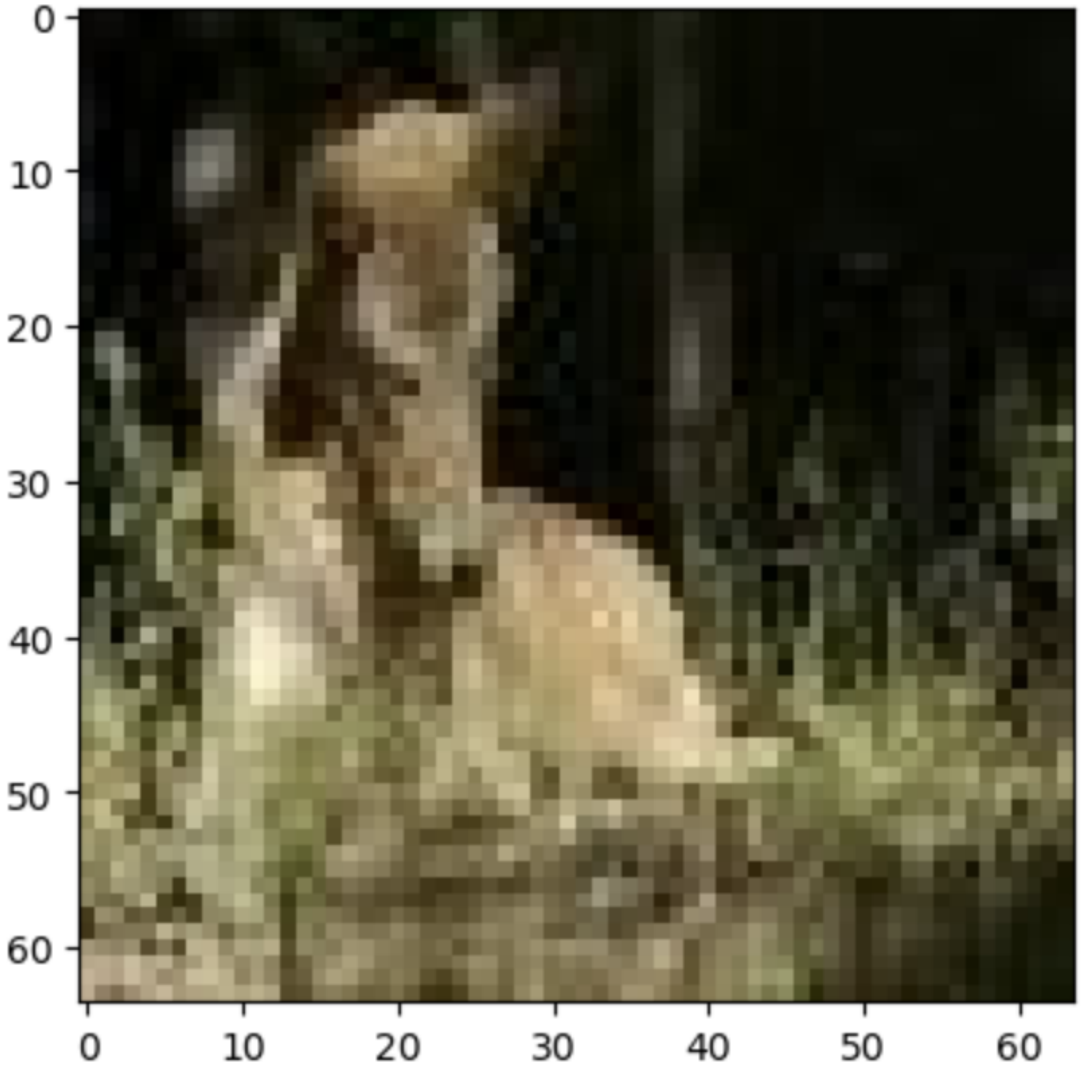

Figure 8: The second image from the ANIMAL-10N dataset.

3.2 Extra Discussion

There appears to be a mislabeling issue with the images in the animal dataset. For example, the image shown below is incorrectly labeled as a hamster, even though it clearly does not depict any animal.

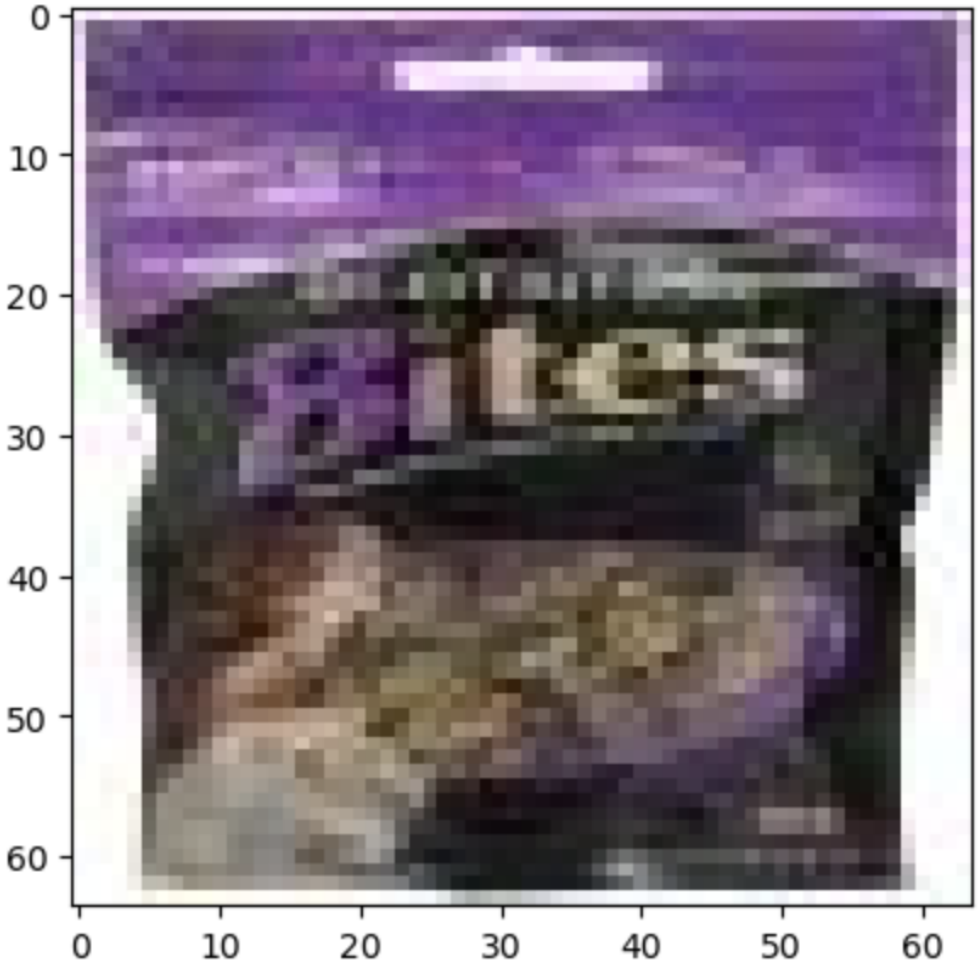

Figure 9: Misidentified image erroneously labeled as a hamster.

Eliminating these incorrectly labeled images could potentially enhance our model's performance. I have encountered several methods that could address this issue. For instance, one approach involves fitting an SVM classifier to extract features from the images to identify anomalies, which are then manually verified Ekambaram et al. (2017).

However, for this lab, we will retain these images to assess the robustness of our model when faced with a noisy dataset.

3.3 Linear model

First, we train a linear model on this dataset using the Adam optimizer with a step size of 0.001. All images were standardized by dividing each pixel value by 255.0, thereby scaling all pixels to be between 0 and 1. The model was initially set to train for 500 epochs, constrained by my computational limitations.

The cross entropy on the training set was 1.96, while the loss on the test dataset was 2.01.

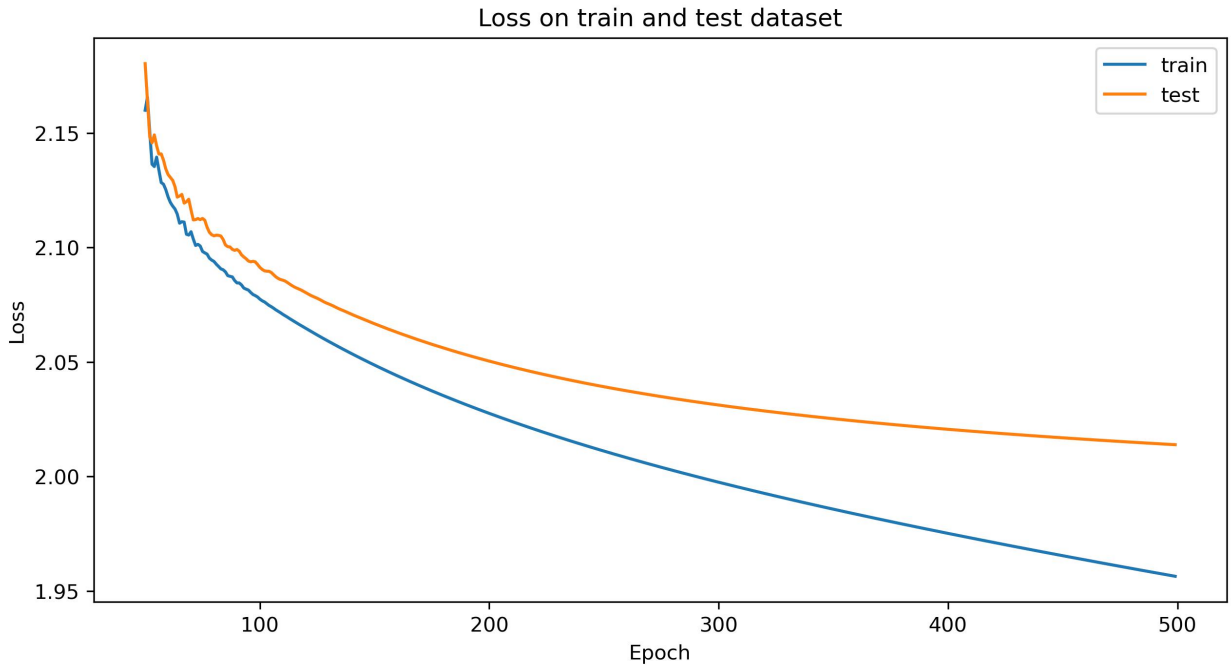

Figure 10: Training and test loss over 500 epochs.

3.4 Vanilla deep convolutional network

Then we fit a vanilla deep convolutional network consisting of a single convolution layer configured as `Conv2d(3, 4, kernel_size=3, stride=2, padding=1)`, followed by a pooling layer with a size of 2, and then a linear layer. And activation is Relu. To accelerate the training process, we employed mini-batching. The model was trained for only 100 epochs due to the slow training speed and sufficient decrease in loss. The loss curve exhibited volatility, which is typical when using mini-batches due to the stochastic nature of the gradient updates. The cross entropy loss on the training set was approximately 1.6, while on the test dataset it was around 1.67.

Figure 11: Loss curves for the training and testing datasets over 100 epochs

3.5 Tune the network

Next, we tune our neural network: the first layer is a `Conv2d(3, 8, kernel_size=3, stride=2, padding=1)` followed by a pooling layer with a size of 2, and the second layer is a `Conv2d(8, 16, kernel_size=3, stride=2, padding=1)` also followed by a pooling layer with a size of 2. A linear layer follows these, and ReLU activations are used throughout. Each hidden layer includes a batch normalization layer. The learning rate is set to 0.001. We do not make this model overly complicated due to the computational limitations of our computer.

To accelerate the training process, we employed mini-batching. The model was trained for only 100 epochs due to the slow training speed and sufficient decrease in loss. The cross entropy loss on the training set was approximately 1.29, while on the test dataset, it was around 1.42, which is much better than previous models.

Figure 12: Loss curves for the training and testing datasets over 100 epochs, showing significant improvements in model performance.

To compare three different models, we plotted the learning curves for these models. We only display the first 100 epochs since we only trained the last two models for 100 epochs, and there was no significant improvement in the linear model after 100 epochs. We also exclude the first 10 epochs from the analysis.

We observed that all models show a decreasing trend in loss, and the performance curves for the training dataset are similar to those for the test dataset. The more complex CNN Model achieves the highest accuracy (lowest cross entropy), followed by the simple CNN Model, and the linear model performs the least effectively.

From the experiments where we tuned the hyperparameters, the learning rate proved to be very important to ensure convergence.

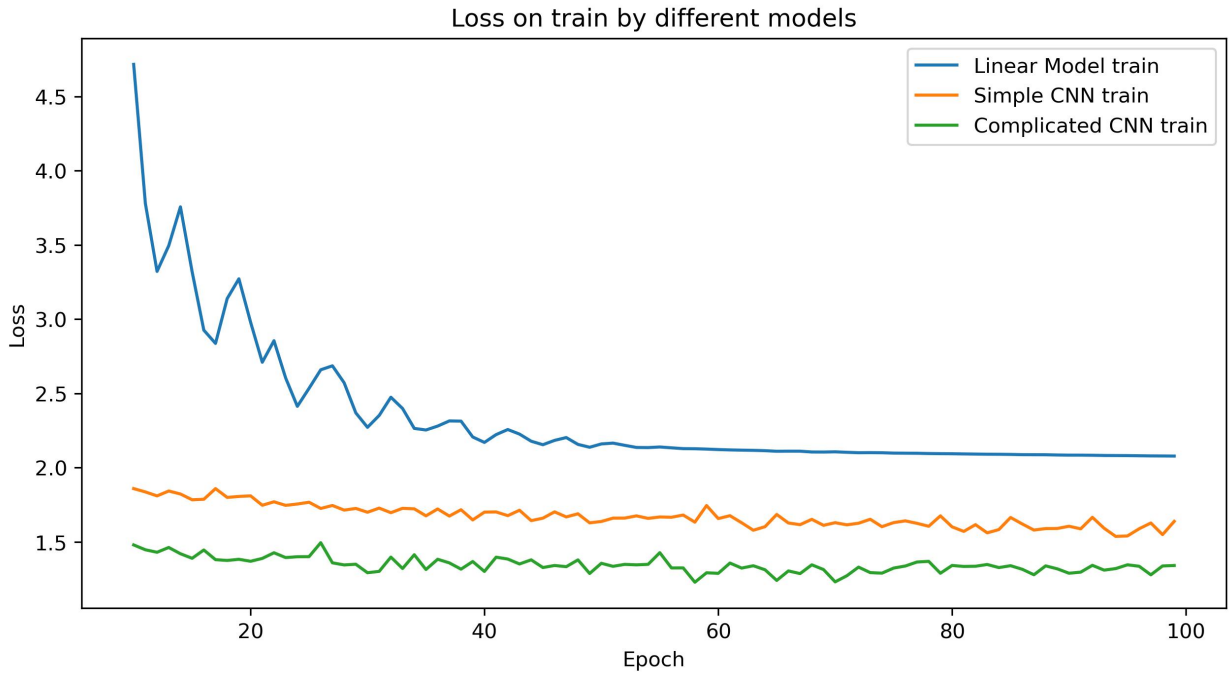

Figure 13: Comparison of learning curves across three different model architectures, highlighting the effectiveness of each model over epochs.

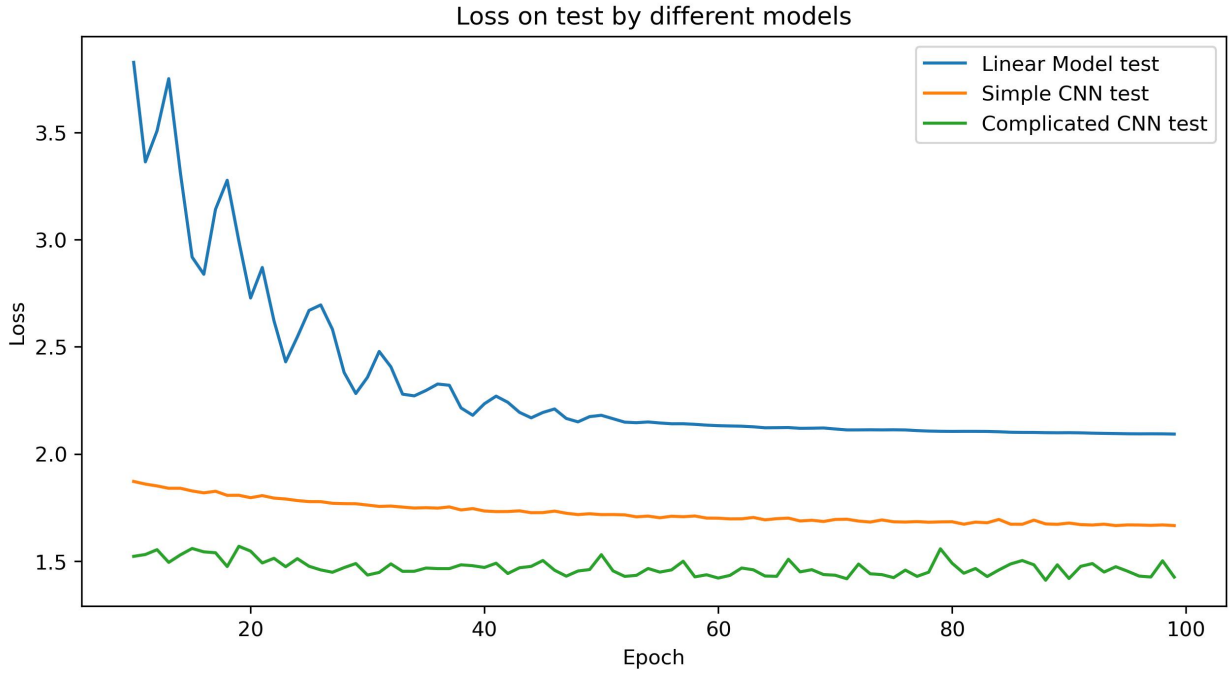

Figure 14: Test dataset learning curves comparing the three models, illustrating the generalization capabilities of each.

4 Feature Atribution on Animal Images

4.1 SmoothGrad

We apply SmoothGrad to the prvious complex convolutional neural network model. The noise level is defined as $\sigma/(\text{xmax} - \text{xmin})$, where xmax and xmin represent the maximum and minimum pixel values of the image, respectively. Using SmoothGrad, we observed that our model is capable of identifying distinct and interesting patterns across different images.

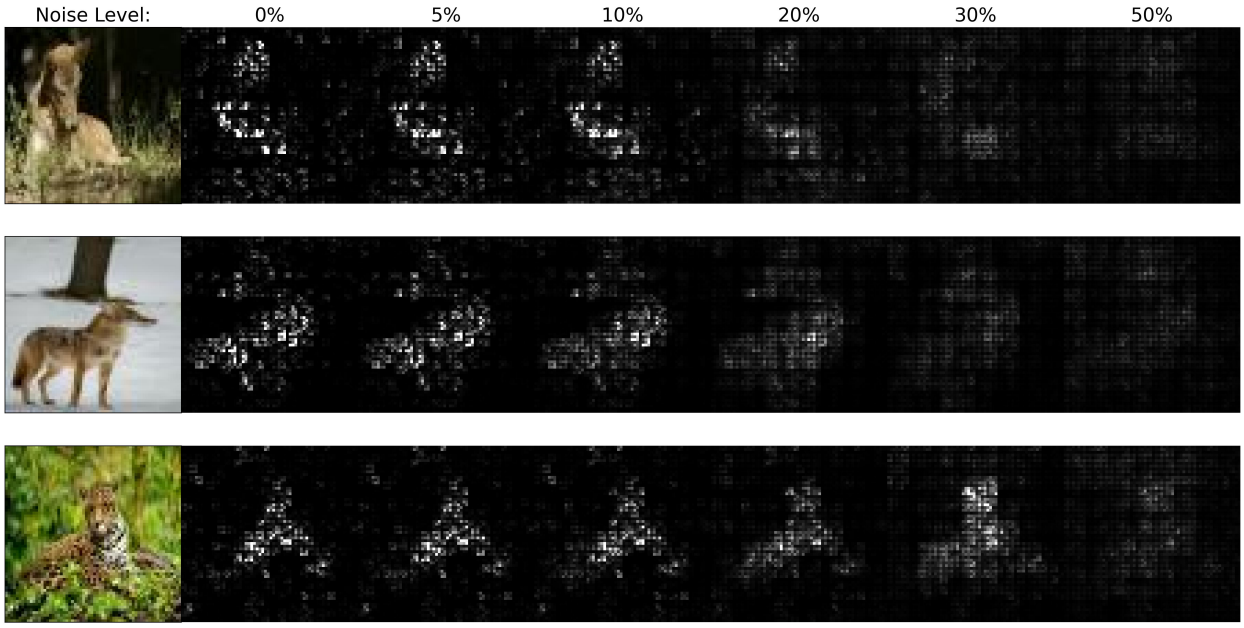

Figure 15: Visualization of patterns identified by the model using SmoothGrad, highlighting different responses for various images.

4.2 LIME

We applied LIME to the same images used in the SmoothGrad section but with our more complex convolutional neural network model. The results were unsatisfactory: LIME only highlighted the boundary for the second image. Interestingly, it appeared to delineate the boundary between parts of an animal and parts of a tree. This outcome suggests that the explanation provided by LIME may not be robust. This could be due to either the model's insufficient ability to discern strong patterns or LIME's limitations in representing the model's characteristics accurately.

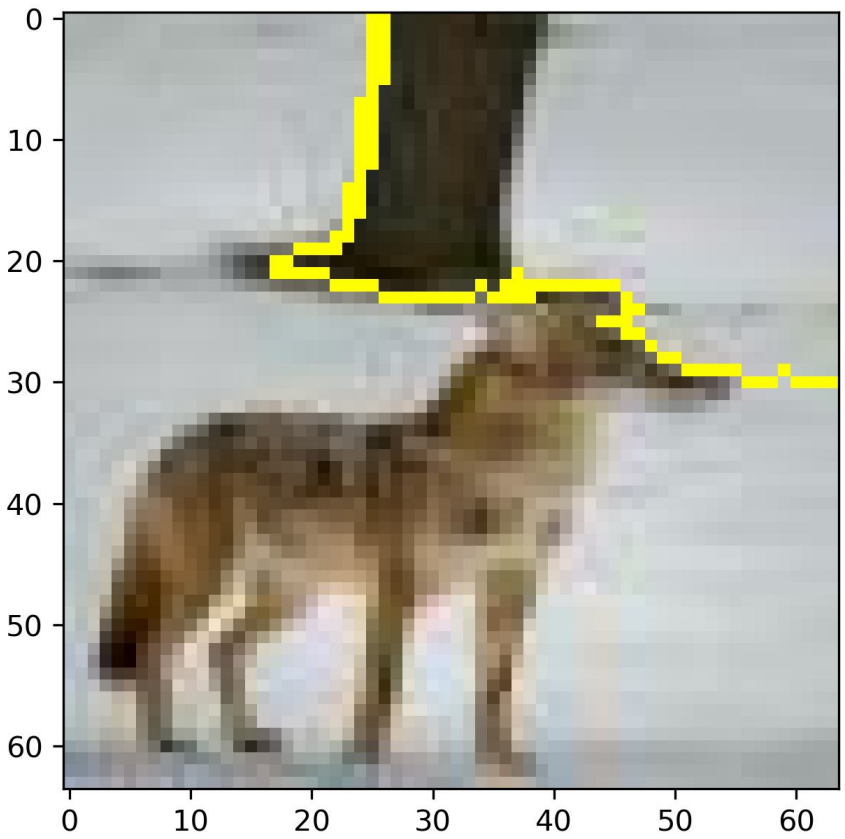

Figure 16: LIME visualization highlighting the boundary detection in the second image, illustrating the model's limited explanatory power.

References

Rajmadhan Ekambaram, Dmitry B. Goldgof, and Lawrence O. Hall. Finding label noise examples in large scale datasets. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2420–2424, 2017. doi: 10.1109/SMC.2017.8122985.