The School of Mathematics

THE UNIVERSITY
of EDINBURGH

Anomaly Detection with Bayesian Neural Networks

by

Theodoros Ladas

Dissertation Presented for the Degree of
MSc in Statistics with Data Science

July 2021

Supervised by
Dr Bruce Worton and Dr Daniel Paulin

Executive Summary

Here comes your executive summary ...

Acknowledgments

Here come your acknowledgments ...

University of Edinburgh – Own Work Declaration

This sheet must be filled in, signed and dated - your work will not be marked unless this is done.

Name: .....
Matriculation Number: .....
Title of work: .....

I confirm that all this work is my own except where indicated, and that I have:

- Clearly referenced/listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc)
- Given the sources of all pictures, data etc. that are not my own
- Not made any use of the report(s) or essay(s) of any other student(s) either past or present
- Not sought or used the help of any external professional academic agencies for the work
- Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
- Complied with any other plagiarism criteria specified in the Course handbook

I understand that any false claim for this work will be penalised in accordance with the University regulations (`https://teaching.maths.ed.ac.uk/main/msc-students/msc-programmes/statistics/data-science/assessment/academic-misconduct`).

Signature .....

Date .....

Contents

List of Tables

List of Figures

1 Introduction

1.1 Motivation

Anomaly detection is a very useful technique, leveraged (among others) by the banking sector in order to automaticaly block fraudulent transactions. This dissertation project is a detailed explanation of how an anomaly detection system could be implemented using Bayesian neural networks. This application, uses machine learning to train a model on a set of data, in order to predict an outcome of interest. Afterwards, the predictions are drawn many times (bootstraped), in order to create an estimation of confidence about the prediction. The goal of the project is to automatically flag transactions that could be fraudulent, thus allowing a human to spend time in deep investigation of those transactions, rather than manually going through each transaction one by one.

This report is divided into the *Introduction*, *Exploratory Data Analysis*, *Methods*, and *Results* sections and a brief summary of each one is presented here. The rest of the *Introduction* section will explain the specific dataset used for the presentation of the problem, as well as all the software requirements to be able to reproduce the results. Next, the *Exploratory Data Analysis* section, presents a various aspects of the data, such as the distributions of the features, their corelations as well as a quick way to visually discover potential anomalies. On the *Methods* section, the specific architecture of all the neural netwokrs models are presented, as well as the way the best model was selected (model validation). Finaly, three related but different metrics are presented in the *Results* section of what constitutes as an anomaly, yet the specific threshhold for those metrics, is arbitrarily set. In a real life application this is a business decision, that should be set by the stakeholders according to their criteria.

1.2 Data Sources

The dataset used in this project, was given by the University of Edinburgh in the context of the dissertation project and it is the wine dataset. It is an especially clean and very well known real-life dataset for prototype creation. The basic assumption is that if the algorithm manages to capture anomalies on this dataset, it is in principal possible to productionise a variant of the architecture for an application of interest. More specifically it is a 4898 (rows) $\times$ 12 (columns) matrix, with no missing values and no duplicated rows. Each row represents one wine and the columns are the various features of each wine, such as its degrees of *alcohol*, the *acidity* of the wine, the *pH* level wich measures how acidic or basic the solution etc. The problem this dissertation is going to try to solve, is a classification problem, of the *type* of the wine, according to all other features. The variable *type* is either *1, 2, or 3* and the classes are almost perfectly balance thus making easier the preprocessing stage of the problem, and focusing more on the architecture and the interpretation of the algorithm as well as its results. Thus, the only preprocessing steps needed are the centering and scaling of the data matrix, to ensure that variables can potentially have the same impact regardless of the scale they are measured in and one-hot encoding the *quality* feature, since its also not a numeric variable, but a factor one. Lastly, The dataset is split into train (80%), validation (25%) and test (25%) sets to ensure that no algorithm overfits the given dataset.

1.3 Software

A combination of the programming languages R and Python are used to produce this report. The use of both languages is in no way binding. R is used for the exploration of the dataset, as well as for the dimensionality reduction plots, while Python is used in combination with tensorflow,tensorflow-probability, and keras to build, validate, and test the neural networks. These packages can also be used from the R ecosystem, yet the setup process is more complicated and thus avoided. The reason R is selected for the EDA, is due to the ggplot2 package, wich is a very powerfull and easy package for creating complicated plots.

2 Exploratory Data Analysis

In this section a basic overview of the dataset and its properties is going to be presented. Firstly, various statistics regarding the dataset are presented in the form of graphs, in univariate, bivariate and multivariate analysis. Afterwards, the dataset it reduced in dimensions with two techniques (a linear and a non-linear) in order to be able to plot it with a two-dimensional graph. This is also a quick way to identify potential anomalies as well visually. The reason of the extended EDA, is to understand what preprocessing steps could be needed in order for the future models to work correctly. Also, the knowledge gained from this exploration of the dataset will help in the explanation of the model in the *Model Validation* section.

2.1 Univariate Analysis

The dataset, consist of twelve feature variables, out of which only one (*quality*) is categorical, while all the others are numeric. The target variable (*type*) is also a target variable with three levels type=1, type=2, type=3. There are no missing values.

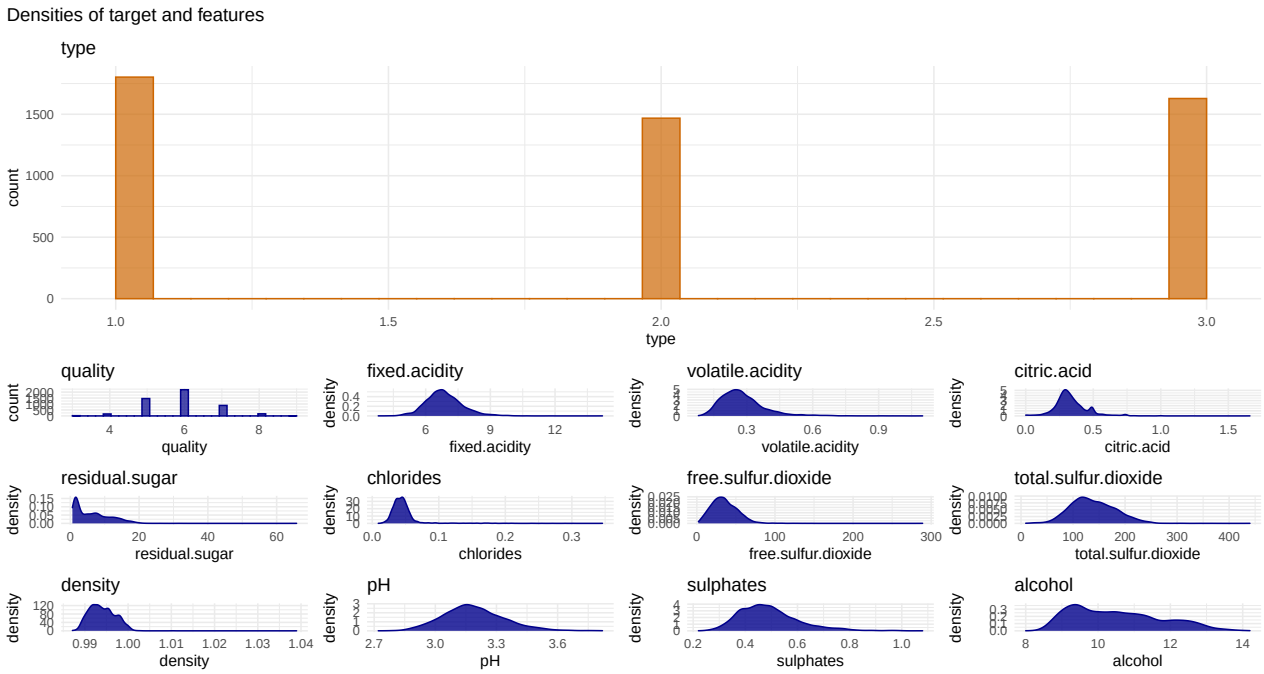


Figure 1: centered image

In Figure 1, the target variable type is represented by the orange barplot. The number of rows in each class is balanced therefore no preprocessing to upsample (downsample) the underepresented (overrepresented) classes is needed. On the variable *quality*, the majority of cases are $(quality \geq 5) \vee (quality \leq 7)$, which would make the very poor and very good wines difficult to predict. Finally, evidently the features, are measured in vastly different scales. For example pH is ranging from 2.7 to 3.6, while *free.residual.dioxide* is ranging from 0 to 300. This means that centering and scaling the data matrix is crusial for any algorithm to work properly.

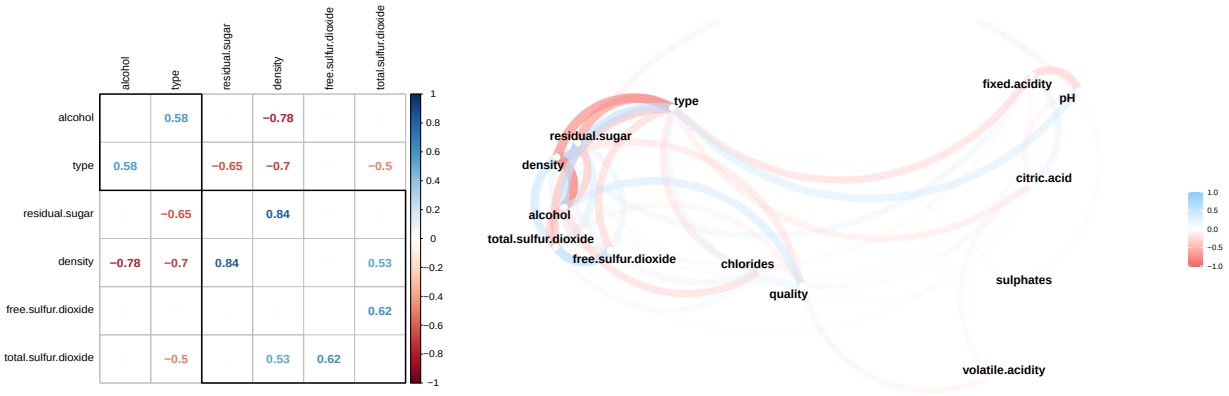
2.2 Bivariate Analysis

Extending the univariate analysis of the previous subsection, the bivariate analysis reveals how the features correlate to each other. Figure 2 (a) and Figure 2 (b) are both producing the same information but presented differently. On Figure 2 (a) the exact values of the linear relationships are shown, while

on Figure 2 (b) clear clusters of high and low linear correlation are formed. Out of these clusters that are present in Figure 2 (b) some are expected but others are not.

For measuring the correlation, the *Pearson's correlation coefficient* is used, which measures the covariance of two random variables, X and Y, and normalizes it with the product of their standard deviations, or:

$$\rho(x, y) = \frac{\text{cov}(x, y)}{\text{std}(x)\text{std}(y)} \tag{2.1}$$



(a) Correlation matrix

(b) Correlation network

Figure 2

For example, Figure 2 (a) shows a strong positive relationship between *free.sulfur.dioxide* and *total.sulfur.dioxide*, which is expected, as *total.sulfur.dioxide* is the *free.sulfur.dioxide* plus other sulfur dioxides from other ingredients. Intrestingly, the same is not true regarding *volatile.acidity*, *fixed.acidity* and *citric.acid*. The feature with the most strong correlations however, is *density*, as it strongly correlates, both positively or negatively with *alcohol*, *type*, *residual.sugar* and *total.sulfur.dioxide*.

In general we identify two clusters. One with strong correlations containing *type*, *residual.sugar*, *density*, *alcohol*, *total.sulfure.dioxide* and *free.sulfure.dioxide*, and a second one with low correlations containing *fixed.acidity*, *pH*, *citric.acid*, *sulphates* and *volatile.acidity*. The model is going to leverage this relationships in order to predict the outcome.

2.3 Dimensionality Reduction

In the exploration of the dataset, two different techniques were used in order to reduce the dimensions and visualize it with plots. The first method is the Principal Component Analysis (PCA), which a singular value decomposition (SVD) of the centered, data matrix. The SVD of an $m \times n$ matrix A is the factorization of the matrix into $USV^\top$. Secondly, a more advanced nonlinear dimensionality reduction method, called t-distributed Stochastic Neighbor Embedding (t-SNE) was tryied, to cross evaluate the results of PCA. This methods works by finding a way to project the high dimensional data into a lower dimension, while preserving the culstering of the higher dimension.

2.3.1 PCA

Using the singular value decomposition to factorize the data matrix X, we produce, three matrixes, U, S and V, each carrying information about the dataset in some aspect. First of all, we can create the following graph of the percentage of variance explained by each principal component.

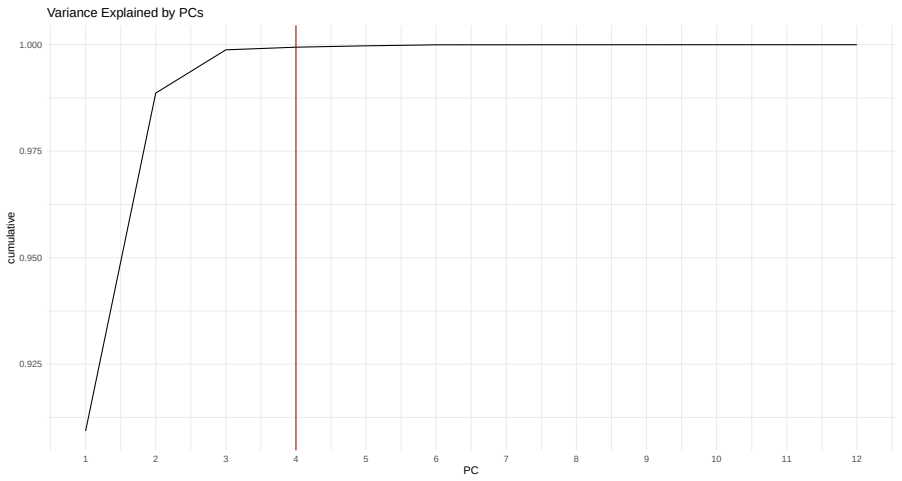


Figure 3

In Figure 3, we can see the cummalative variance explained by each additional principal component. This is a very important graph as we can reduce the dimension of the data matrix by selecting a threshold (for example, 99% of the variance). In this example, it is clear that with only the first four principal components, that threshold is surpassed.

The principal components are a linear compination of all the columns of the dataset, and each one is perpandicular to the previous one. We can continue the exploration by visualizing the weights (loadings) of these four principal components and try to understand wheather these components make sense according to our knowledge on the topic.

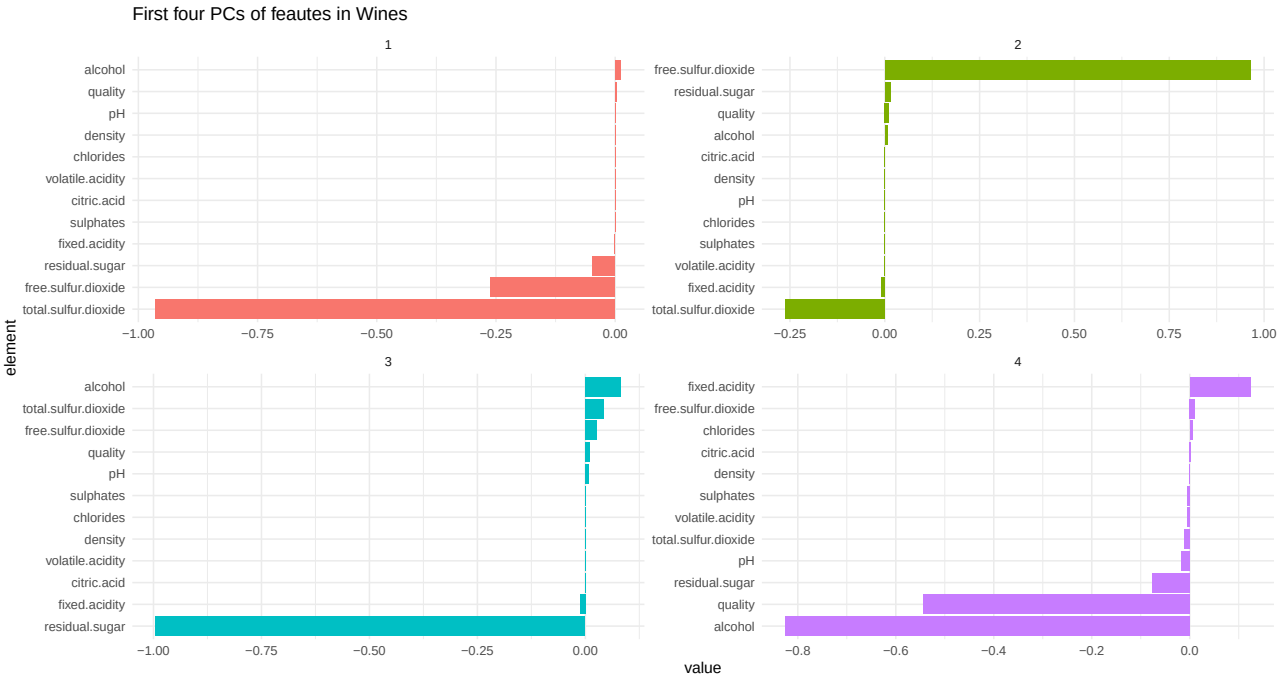


Figure 4: centered image

In **??** the first four principal components are visualized along with the loadings of each columns. By on the two most extreme positive and negative values of the loadings, we can interpret the components into new kinds of variables.

- PC1: Alcohol vs Sulfur Dioxide (free and total)

- PC2: Free Sulfur Dioxide vs Total Sulfur Dioxide
- PC3: Alcohol vs Sugar
- PC4: Alcohol vs Acidity

It is clear that this dimensionality reduction technique produces results that reflect the real word. Sulfure Dioxide is a vital component in wine making as it regulates bacteria growth among other important tasks, yet, it also gives unpleasent oddors and tastes to the wine. PCA immidiately captures that reality by assigning the two biggest negative loadings on sulfure dioxide concentration (both free and total) to the first pricnipal component. In addition to that, the second most important component in order to classify the wines, is the origin of that Sulfure Dioxide. The total Sulfure Dioxide is the Free Sulfure Dioxide plus Sulfure Dioxide that is bound to other ingredients such as sugars etc. After investigating how much SO2 a wine has, as well as were it comes from (free vs total), the next most important factors have to do with the specific taste of the wine, mainly how sweet and how acidic the taste the wine has.

We can now visualize the first two pricnipal components on a plot, while also coloring the datapoint according to their *type*, the target variable. It is immidiately clear by Figure 5 that *type* classes two

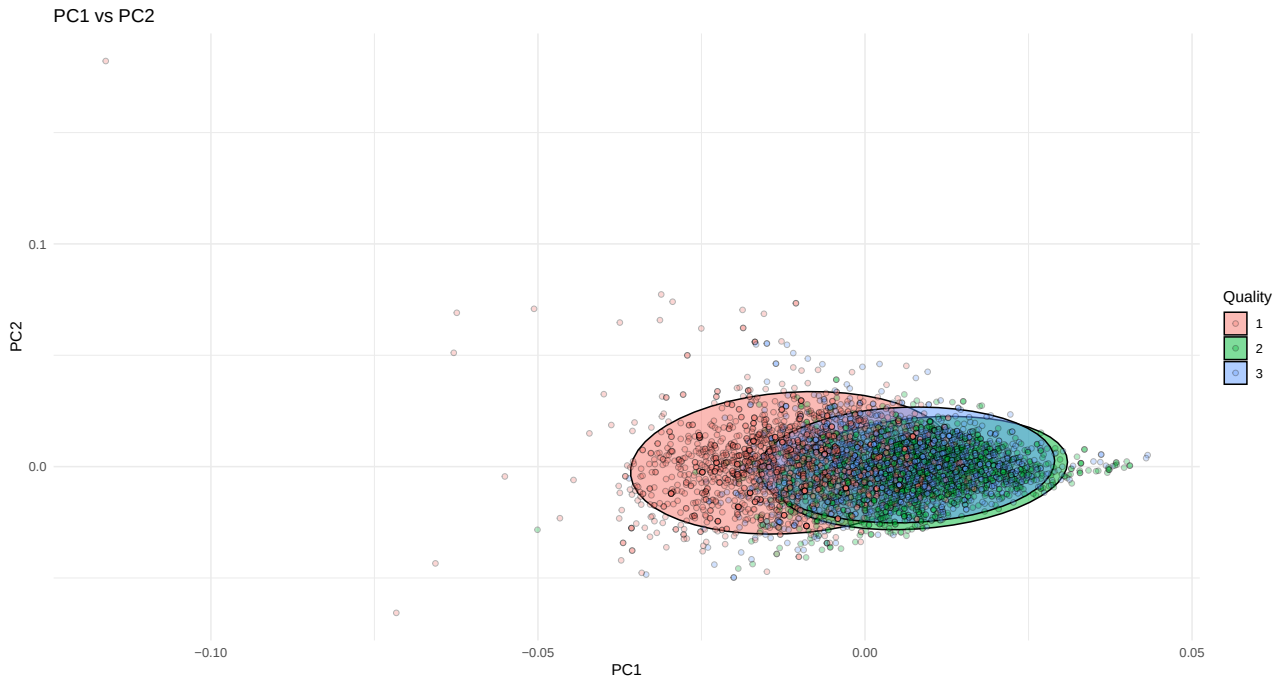


Figure 5: centered image

and three, have an extensive overlap in this graph. That is a useful information in the model evaluation, as we expect the model to "find it difficult" to distinguish a wine of type two from a wine of type three, while simultaneously having better accuracy in predicing class one. However, there a second major takeaway from *Figure 3*, and that is the existance of potential anomalies. While the majority of the points are centered around $(0, 0)$, some points are geometrically far away from the others of the same class.

2.3.2 t-SNE

Principapal Component Analysis through singular value decomposition produced important findings and deepened our undestanding of the dataset. However, it was clear by *Figure 3*, that it could not produce a clear enough separation between the classes. This could be because of the nature of the

dataset, or it means that PCA is too simple to capture the complexity of the dataset, due to its linear nature.

Therefore, a second, non-linear technique was tried in order to produce the corresponding figure as *Figure 3*. This technique, is based on calculating the distances from each point to each neighbords and imposing a t-distribution on those values. It is an iterative algorithms that many times produces very good clustering. However, in Figure 6, it is clear that *type* classes two and three are still clustered together, while class one, is separated.

The main takeaway from this analysis is that the dataset, although clean and simple in nature to understand, is mode complicated than it looks. This suggest that a simple model such as logistic regression might not be the model of choice. Neural Networks are a good candiate for such task due to their flexible nature.

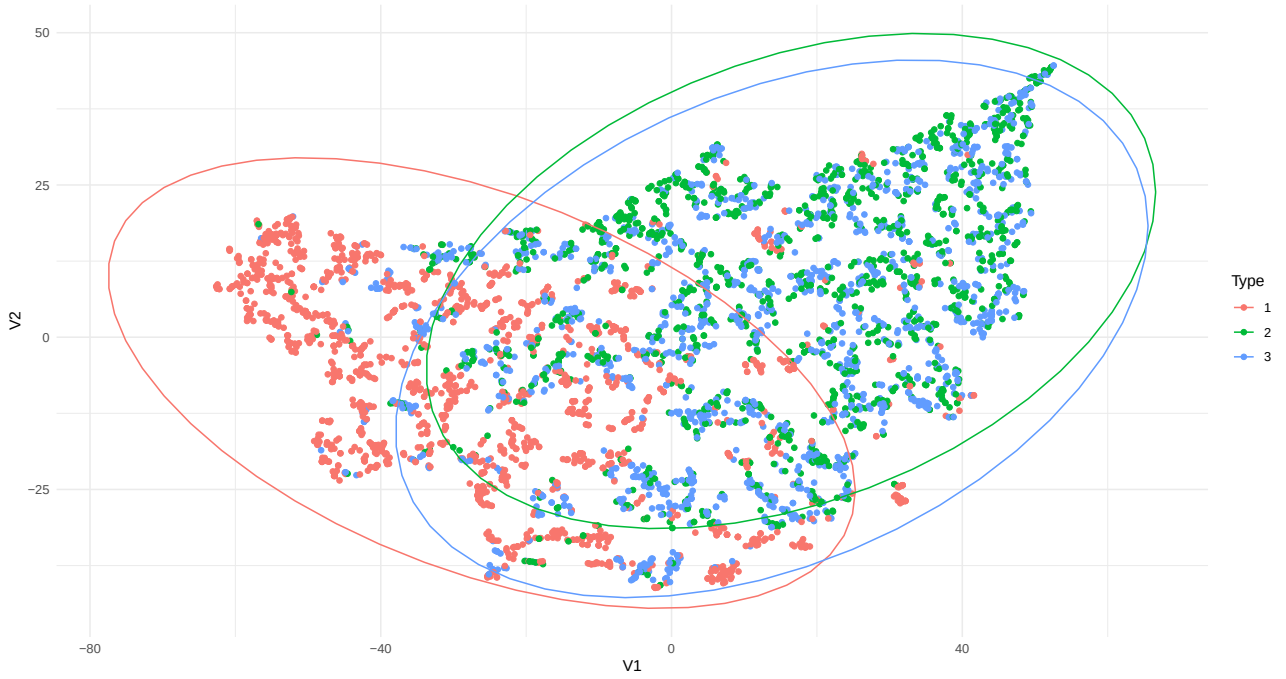


Figure 6: centered image

2.4 Multivariate Analysis

With the information revealed from PCA, we can now plot the highest and the lowest loadings of the four principal components, for each level of the target variable. The reason for plotting this graph is to investigate wheather some level of the *type* variable, behaves different that the others when plotted on the same axis.

In all four subplots of Figure 7 we identify the same exact behaviour. The coefficient of the regression is negative on the *total.sulfure.dioxide* against *alcohol* axis for all levels, positive on the *total.sulfure.dioxide* against *free.sulfure.dioxide* axis, and close to zero for the other two PC scores. This finding, while not very exciting, is important, because it confirms once again that the relationship of our target variable *type* and the features identified at the dimensionality reduction stage (*total.sulfure.dioxide*, *free.sulfure.dioxide*, *alcohol*, *residual.sugar*, and *fixed.acidity*) are important for establishing a good model.

Figure 7

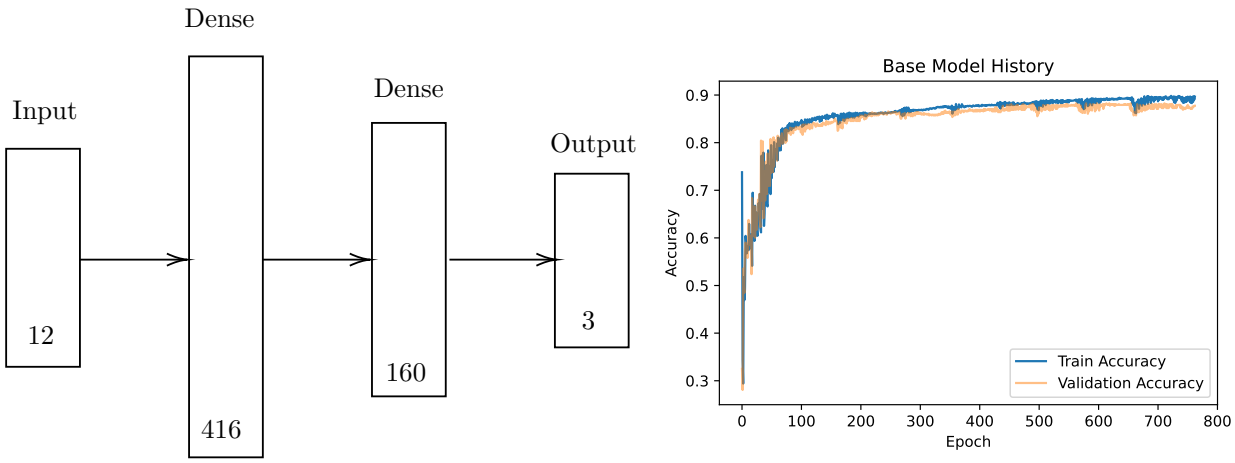
3 Models

In this section, the methods used to create the model will be discussed. This includes, the baseline model that is deterministic in nature, as well as the various Bayesian approaches. Firstly, the reason to choose neural networks as the model of choice, is due to the complexity of the research question, in combination with the complexity of the dataset as demostrated from the exploratory data analysis. Secondly, the Bayesian approach is important to make the neural network prediction *stochastic*.

In order to train the model and produce both high accuracy on the training and validation sets, as well as generalize well on the test set, a series of actions had to be taken. Firstly, a set of hyperparameters had been set for all models. Those hyperparameters are:

- The Learning Rate - ranging from $1e-1$ to $1e-8$
- The number of layers between the Input and the Output - ranging from 2 to 8
- The activation function of these layers - where the choices where:
  - Rectified Linear Unit (reLU): $f(x) = \max(0, x)$
  - Exponential Linear Unit (ELU): $f(x) = \max(e^x - 1, x)$
  - Sigmoid: $f(x) = \frac{1}{1+e^{-x}}$
- The number of units in each function - rangin from 32 to 512, with a step of 128

The optimization process, was a RandomSearch algorithm, implemented with the KerasTuner package. This results in slightly different results in every run, however by switching to a GridSearch algorithm instead, the same outcome can be ensured each time. However, this process takes more time, since it is searching the whole hyperparameter space, while the RandomSearch takes less time to find the best model in each run. The metric used to judge which model is best was chosen to be *accuracy*, due to its simplicity and suitability at the research problem. Lastly, the loss function that the models are minimizing in order to produce the best weights, was a custom made function that calculated the sum of the categorical crossentropy between the true and the predicted labels. The true labels have a known (for the traingin set) probability distribution (PMF), and the predicted labels, also have their own probability distribution (PMF). When the sum of the categorical crossentropies is



(a) Base Model Neural Network Architecture

(b) Base model History of epochs

Figure 8: Baseline Architecture and Epoch History

minimum, it shows that the predicted PMF and the true PMF are as close as possible. Mathematically this is expresses as:

$$\text{Loss} = \sum_{i=1}^{3} y_{true}^{(i)} \log \frac{1}{y_{pred}^{(i)}} = -\sum_{i=1}^{3} y_{true}^{(i)} \log y_{pred}^{(i)} \tag{3.1}$$

Lastly, the models are trained for 2000 training epochs. after a certain number of epochs, started to overfit. In order to avoid this pitfall, early stopping was implemented in the model with a patience of 200 epochs. That means that if the validation accuracy, drops for 200 epochs, the model assumes that it overfitted and stops trying to further minimize the loss function. This is why the x-axis varies in every Figure (**??**, **??**, **??**) where the history of accuracy per epoch is plotted.

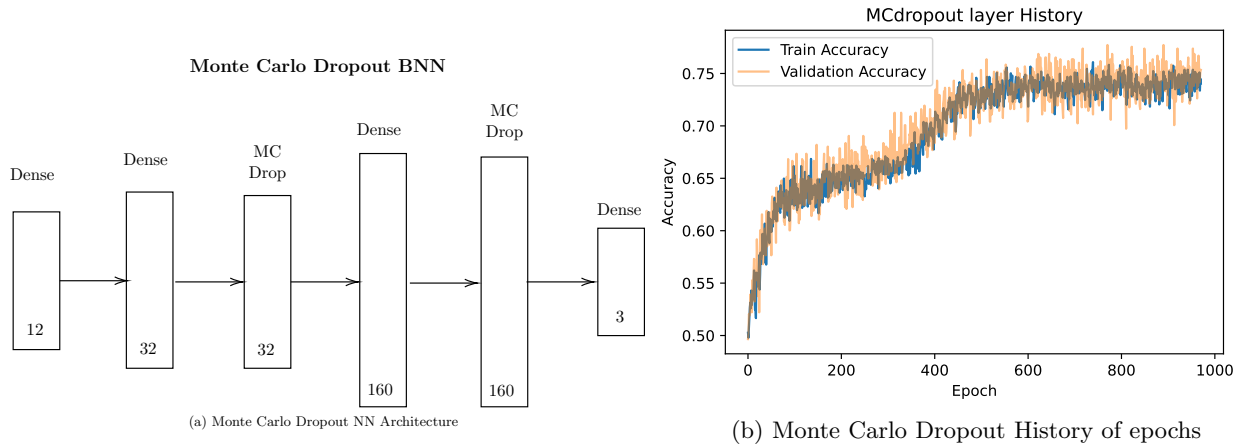
After the sucessful training of the base neural network, the predictions are deterministic. That means that each time a specific input is passed through the NN, the same output is produced. While this can be useful in certain applications, the goal of this project is to find a metric for the *uncertainty* of the predictions. Thit is impossible to do with a deterministic process as samples cannot be drawn from the posterior distribution to estimate it. Therefore, after establishing a baseline, to prove that the problem can be solved with good enough accuracy, the network can become Bayesian by changing some of its layers to Bayesian layers. Afterwards, we can sample from the posterior distribution of the predictions, by predicting n times the same input, in our case the test set. Then by storing each predicted probability (one for each class), in each draw $(1 \dots n)$ we can compute average predictions, variances, confidence intervals, and more metrics discussed in section 4.

3.1 Base Neural Network

Before tackling the Bayesian Neural Network, a deterministic baseline should be first implemented. The model is built using the keras package for Python. The architecture the produced the best results is one with two hidden layers of 416 and 160 neurons respectively, densely connected to one another. This produces a total number of parameters to be estimated of $75,107$. The neural network is architecture depicted in **??** (a).

?? (b) is a typical history of the training accuracy increasing over each epoch. Typicaly it stops before iteration 1000 and the best accuracy of the model is close to 90%. This is a very good accuracy, since as we have seen at the exploratory data analysis, *type 2* and *type 3* are very difficult to distinguish.

With the baseline model explained and trained, we gained an understanding at the extend to



(a) Monte Carlo Dropout NN Architecture

(b) Monte Carlo Dropout History of epochs

Figure 9: Monte Carlo Dropout Architecture and Epoch History

which the problem can be solved with neural networks. Arguably, an accuracy score of 90% does not mean anything in a vaccum. 90% accuracy in predictin a cetrain desease in a patient might be too low, while the same accuracy on an application less sensitive to important decision making might be exelent. However, in the specific task at question (classifying types of wines), this could be a good final score for the model.

3.2 Monte Carlo Dropout

Adding some Monte Carlo Dropout layers in the middle of each Dense layer, is the first approach in turning the base model into a Bayesian one. Simple dropout layers work during the training of the neural network, by randomly ignoring or "dropping out" a percentage of neurons along with the backward and forwand connections. This is being handled by random draws of a Bernoulli distribution with the "success" probability defined by the user. That process, still generates a *deterministic* prediction, since at test time, which neurons are online is already decided. Monte Carlo dropout, is a wrapper of the simple dropout funciton that preserves the dropout at testing time. That way, the same prediction passed through the network twice, will produce different predictions according to which neurons and connections are online. Finally, these predictions are interpreted as random draws from the posterior distribution.

It is clear in **??** (b) that there is a lot more variance in the prediction from epoch to epoch due to the random nature of the dropout layers. An important finding in this approach, is that the validation and the training accuracy are almost in all epochs at the same level. Contrary to the base model, where the validation accuracy increases with a slower rate than the training accuracy, this does not happen when the Monte Carlo Dropout layers are added to the model. It can even maintain for a short duration a higher validation accuracy than training accuracy, as seen from epochs 200 to 400 in **??** (b) due to random chance.

3.3 Other Bayesian Approaches

In addition the Monte Carlo Dropout model, various other approaches have been tryied. Their results on training where not as good as the MC dropout model, hence they are presented in this section and their specific architecture is in Figure **??**. A more robust way of validating the models is presented in section 4.

For both the Variational and the Flipout models, a $N(x; 0, 1)$ distribution was choosen, since we have no prior information to choose otherwise. While the Fliptout model behaves similarly to the MC dropout the Variational model, is behaving completely at random. This could suggest that something is wrong with the prior set.

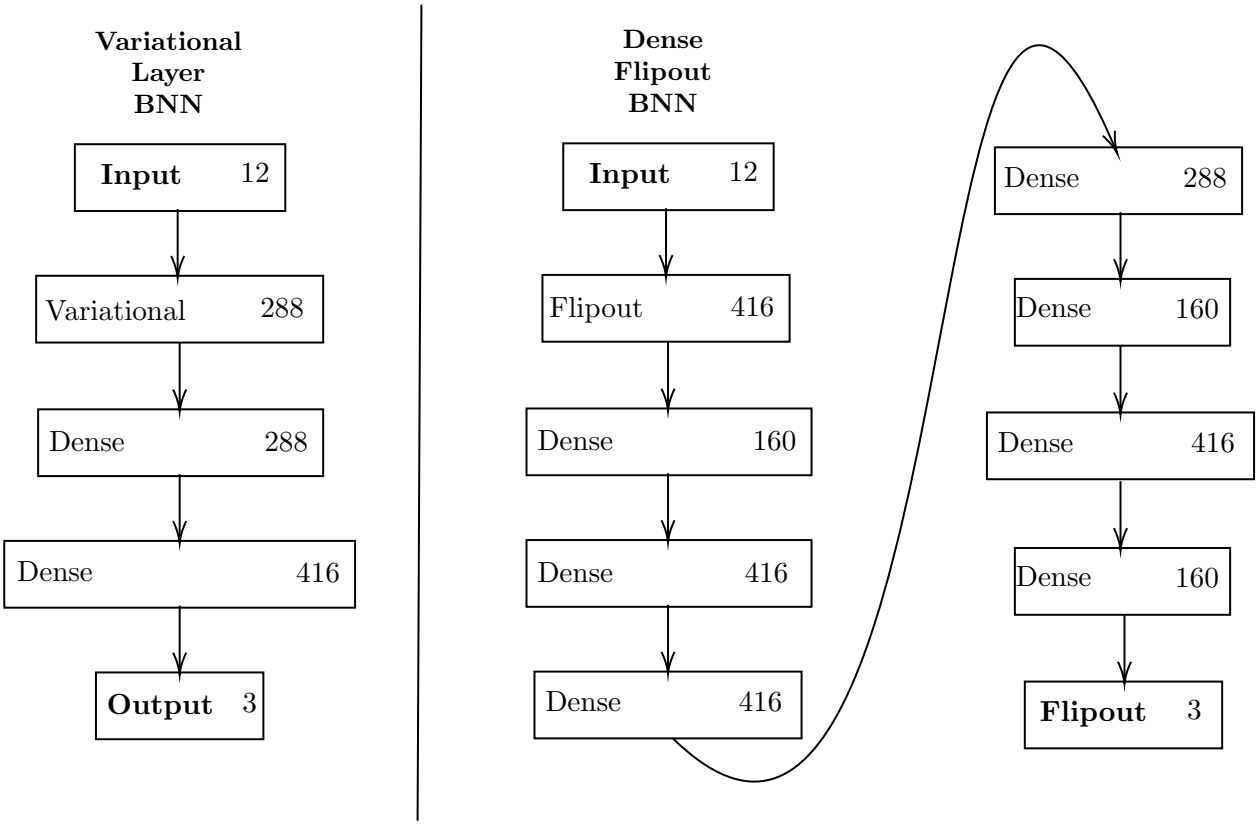
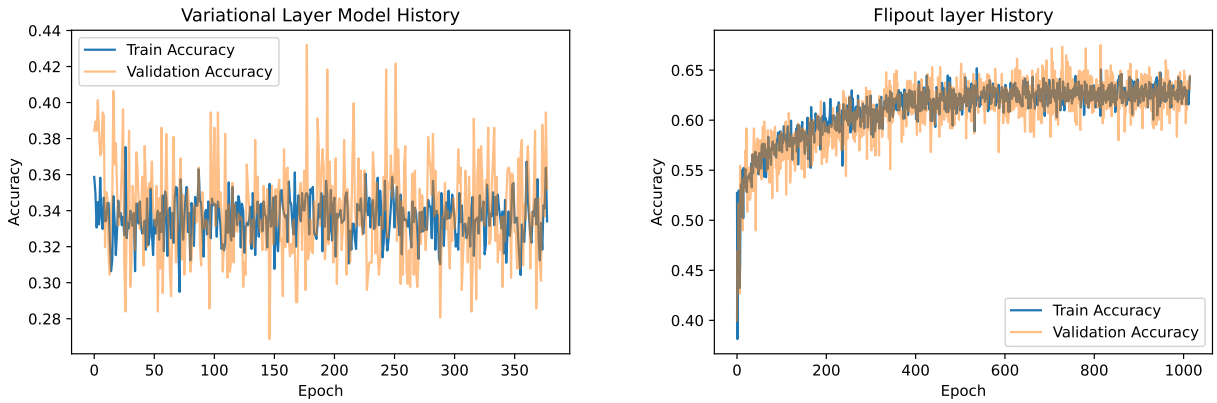


Figure 10: Variational and Flipout model Architectures



(a) Variational layers History of epochs

(b) Flipout layers History of epochs

Figure 11: History of other Bayesian Models (Flipout and Dense Variational)

The Flipout model however, reaches a lower maximum accuracy than the MC dropout model, therefore while it does work as a solution, it is likely not the model of choice.

3.4 Model Validation

In order to choose the best model among the Bayesian approaches the test set is used to calculate the accuracy per class. In Figure 12 the base model (non-Bayesian) confusion matrix is presented. We can clearly see that *type 2* and *type 3* are mixed, as we expected from the EDA.

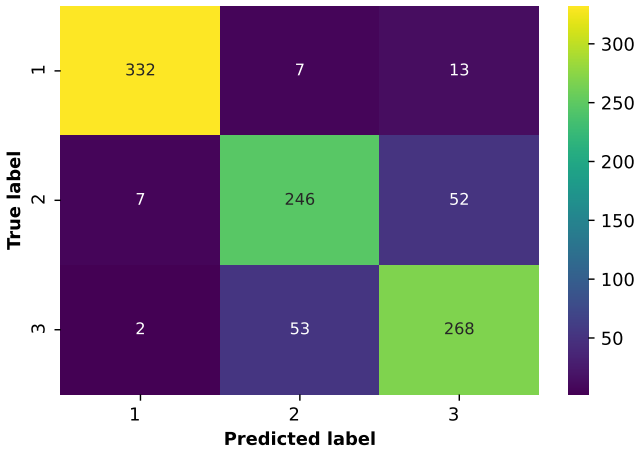
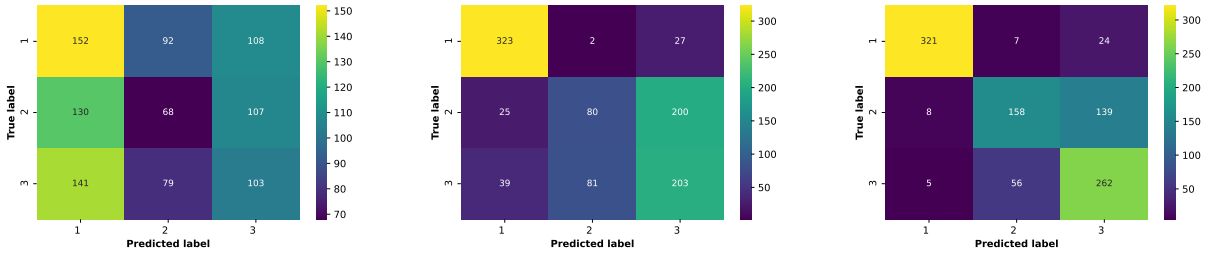


Figure 12: Base Model Confusion matrix

In Figure 13, the same confusion matrix for all the Bayesian models are presented. The Variational model is, as seen from the History graph as well, is almost completely random. The Flipout model, is having an exceptional performance in predicting *type 1*, yet it confuses *type 2* with *type 3*. Lastly, the MC dropout model, as predicted from the History plot, is having the best overall performance.

It is abandontly clear therefore that the model of choice for the specific problem at hand is the MC dropout model as it has the highest accuracy across classes.



(a) Variational Confusion matrix (b) Flipout Confusion matrix (c) MC dropout Confusion matrix

Figure 13: Confusion matrices for all the Bayesian Models

3.5 Model Explanation

In this section a deeper validation of the model is presented. Accuracy is a good enough metric for a lot of tasks, however, when the project at hands needs to be both accurate and interpretable to some extend, further examination of what the model "thinks" when predicting is needed.

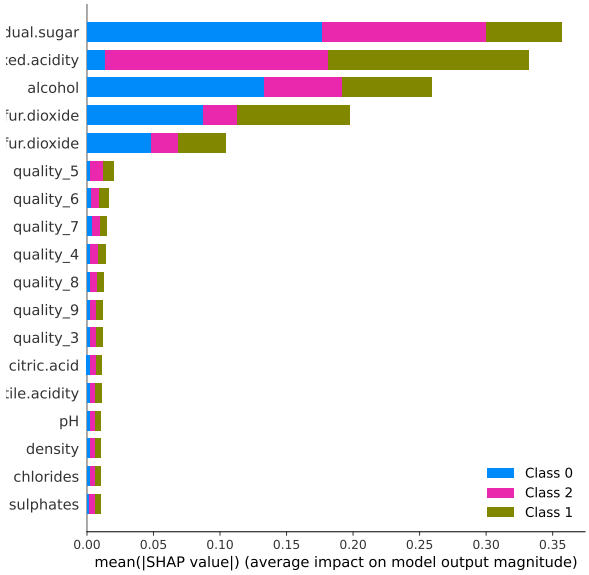
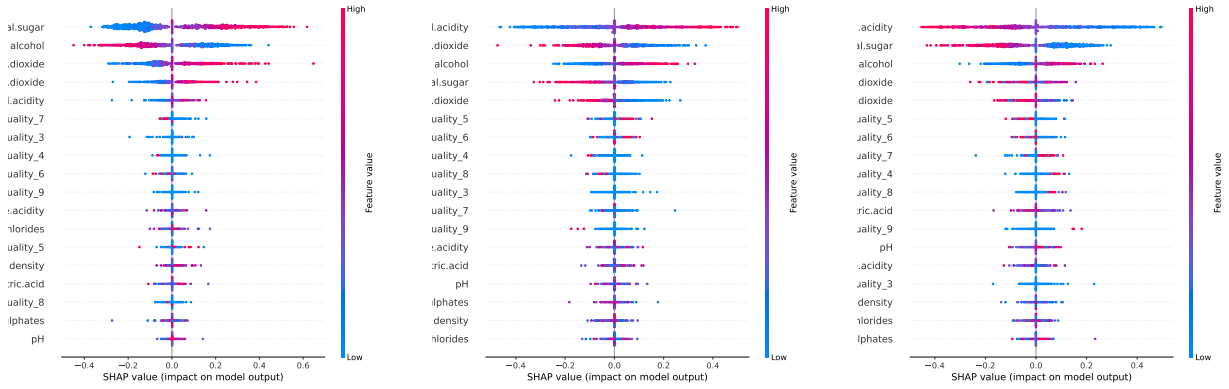


Figure 14: Shapley Values on a feature by feature level

To conduct this analysis, a package called SHAP is used. SHAP stands for *SHapley Additive exPlanation* and is a concept borrowed from Economics. In the Machine Learning field, a Shapley value represents the average marginal contribution of a feature value to the predicted outcome. The next plots are produced on the test set to understand how the algorithm "decides" in a held-out, never seen before dataset.

In Figure 14 we see the following. The top four, most influential variables to the output are *residual.sugar*, *fixed.acidity*, *alcohol*, and *total.sulfur.dioxide*. This seems to be in line with the finding of the much simpler (yet much faster) PCA in the EDA, although the order of the variables is not exaclty the same. More specifically, the mean SHAP value for *residula.sugar* for class zero, is about 0.17, while the same for class two is around 0.13 $(0.3 - 0.17)$ and even lower for class 1. That means that *residual.sugar* is more important when predicting *type 0*, then *type 2* and lastly *type 1*. The same

logic can be extrapolated for *fixed.acidity*, *alcohol*, etc.



(a) Class 1 Explanation (b) Class 2 Explanation (c) Class 3 Explanation

Figure 15: Shap values in a value by value level

Taking a look into one stage deeper into the decision making process of the algorithm, Figure 15 is produced.In Figure 15 (a) we see the most influential features in descentind order.

Firstly, in all three graphs, the SHAP values of all the features are centered around zero. That indicated that the distribution of the test dataset Secondly, as explained in Figure 14, *residual.sugar* is the first most important feature in predicting class zero. However, this graph also provides further granularity. In particular, when the *residual.sugar* values are low, this feature has a low impact in classifying the datapoint as class zero, and opositly, when the value *residual.sugar* then the feature has a very high influence in the "decisition" of the algorithm to classify the datapoint as class zero. The same logic can be extended to the other two classes and the other variables.

While this model explanation did not add anything new to the analysis so far, it is regardgles an important step. Understanding why the model predicts in this specific way and not in some other, is vital for the workflow, as it confirms or rejects our prior beliefs. In this specific case, the prior knowledge gained in EDA seems to be confirmed.

4 Results

The last important task is to produce a measure of uncertainty. This will help to automatically identify datapoints that are anomalous. The assumption is that if the Neural Network is uncertain as to how to classify a specific wine, then that wine needs human judgment to be classified correctly. In order to produce a measure of uncertainty, firstly we need to sample from the posterior distribution. This is being done by drawing $n = 1000$ predictions from the test set. In essense, the test set is being passed through the MC dropout neural network 1000 times, generating a different probability for each class in each pass. Then, the average per class is calculated for the 1000 predictions on each row of the dataset. These are the average probabilities for each class generated by the Bayesian neural network.

In the rest of section 4

4.1 Standard Deviation

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin.

Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

4.2 Confidence Intervals

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

4.3 Entropy

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

5 Discussion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.