Improving Credit Approval Decisions with Explainable Machine Learning

A thesis submitted in partial fulfilment of the University's requirements for the degree of

Master of Science

in

Artificial Intelligence

by

Lawrence Alaso Krukrubo

2039070

to the

School of Mathematics and Computer Sciences

The University of Wolverhampton

Supervised by: **Dr Eva Navarro Lopez**

June 30, 2023

Declaration

I hereby declare as the author, that this thesis entitled **“Improving Credit Approval Decisions with Explainable Machine Learning”**, submitted to The University of Wolverhampton towards the partial requirement of **Master of Science in Artificial Intelligence** is an original work carried out by me under the supervision of Dr Eva Navarro-Ĺopez.

This dissertation adheres to proper academic citation practices by providing clear cross-references to authors, works, and page numbers (where required) for all quoted sentences or passages from external sources. Any illustrations not created by the author of this thesis have been used with explicit permission from the original source, and proper acknowledgement is given.

I understand that failure to do this amounts to plagiarism, and will be considered grounds for failure. Therefore I have sincerely upheld academic ethics and honesty throughout this body of work.

Thank you.

Name: **Lawrence Alaso Krukrubo (2039070)**

Date: June 30, 2023

Acknowledgement

I would like to begin by expressing my heartfelt gratitude and deep appreciation for my mother, who has played the roles of both a mother and a father in my life since my father's passing many years ago. I am immensely thankful for her unwavering support and guidance throughout the years. It is difficult to put into words the profound influence and tremendous support that this remarkable woman has provided me with. Without her sacrifices, discipline, love, and selflessness, I would not be the person I am today. My mother has stood tall and unwavering, fulfilling the roles of both a mother and a father, demonstrating incredible strength and dedication over the course of many years. I wholeheartedly appreciate and cherish you, mother. May God bless you abundantly.

Next, I would like to express my heartfelt gratitude to my Supervisor, Dr Eva Navarro, whose unwavering commitment to excellence has been a constant source of inspiration throughout my Master's journey. Under her guidance, I have developed an interest in Graph-based Data Science and witnessed remarkable growth in my research and academic writing abilities. I must emphasize that Dr Eva Navarro's teachings have been the most influential during my Master's programme. Furthermore, I am truly impressed by her notable contributions to various fields of Science and Engineering.

With profound love, humility, and gratitude, I would like to extend my heartfelt thanks to my uncle, Professor Sammy Dagogo-Jack, for his invaluable contribution to my academic pursuits. I am deeply grateful for his guidance and support throughout my studies. Additionally, I would like to express my special appreciation to Dr. Ibinabo Oboro, Engineer Bekinbo Dagogo-Jack, Mr. Obeleye Krukrubo, Aunt Sisi, my beloved Esther Krukrubo, as well as friends and family members who have continuously encouraged and supported me on this journey. God bless you all

Above all, I wish to thank the Almighty God for the blessings of life, health and faith. In Jesus name. Amen

Lawrence Alaso Krukrubo

Contents

1 Introduction **1**
- 1.1 Problem statement . . . 1
- 1.2 Motivation . . . 3
- 1.3 Research aims and focus . . . 4
- 1.4 Structure of the project . . . 4

2 Literature review **6**
- 2.1 Background . . . 6
  - 2.1.1 AI overview . . . 6
  - 2.1.2 Credit approval overview . . . 9
- 2.2 Key concepts . . . 14
  - 2.2.1 Explainable Artificial Intelligence (XAI) . . . 14
  - 2.2.2 Local Post-hoc Explanations . . . 20

3 Contrastive Explanation Method **22**
- 3.1 Causal explanations . . . 22
  - 3.1.1 Modelling causal explanations . . . 24
- 3.2 Contrastive explanations . . . 27
  - 3.2.1 Limitations of counterfactual explanations . . . 29
- 3.3 Contrastive learning . . . 30
  - 3.3.1 Contrastive problem formulation: . . . 32
- 3.4 Contrastive Explanation Method (CEM) . . . 33
  - 3.4.1 CEM Structure . . . 33
  - 3.4.2 CEM Overview . . . 34
  - 3.4.3 Pertinent Negatives (PN) . . . 37
  - 3.4.4 Pertinent Positives (PP) . . . 38

4 Methodology **39**
4.1 Exploratory Data Analysis (EDA) 39
4.1.1 The data: 39
4.1.2 Feature selection: 41
4.1.3 Model training and evaluation: 42
4.1.4 CEM Explanations: 43
4.1.5 Pertinent Negatives (PN) Algorithm 45
4.1.6 Pertinent Positives (PP) Algorithm 46
4.1.7 Findings & Insights 47
4.1.8 Interactive Visualization: 48

5 Conclusion **49**
5.1 Discussion 49
5.2 Limitation 50
5.3 Future Work 50

Appendices **59**

A Response from ChatGPT **60**

List of Figures

2.1 Major Subfields of Artificial Intelligence . . . 6
2.2 The Transformer Architecture . . . 9
2.3 Sigmoid Curve . . . 11
2.4 Simplified Loan Application Scoring Decision Flowchart . . . 13
2.5 Cross section of global and local XAI techniques . . . 21

3.1 The Causal Model DAG . . . 26
3.2 The Contrastive Representation Learning Framework . . . 32
3.3 Non-Parametric Classification Loss . . . 33
3.4 CEM versus LRP and LIME on MNIST . . . 36
3.5 PN Optimisation Function . . . 37
3.6 PP Optimisation Function . . . 38

4.1 A few categorical variables with multiple categorical values... . . . 42
4.2 The model training metrics: yellow=val, blue=train curves . . . 43
4.3 CEM installation steps using Anaconda... . . . 44
4.4 Feature Importance & PP Scores: Person 57 . . . 48

A.1 What is ChatGPT . . . 60

List of Tables

2.1 Common Factors Affecting Credit Score 11
2.2 European Commission's Key Requirements for Trustworthy AI Systems 15
2.3 Popular Explainable AI Techniques 19

3.1 Fact & Foil: Literary Award 24
3.2 Fact & Foil: Credit Loan 29

4.1 Cross-section of Credit-One Bank Loan Data Variables 40

Abstract

This project aims to enhance credit approval decisions by leveraging the IBM Contrastive Explanation Method (CEM). Credit approval decisions play a crucial role in financial institutions, as they determine the eligibility of individuals to access credit and financial resources. However, traditional credit approval models often lack transparency and interpretability, which can lead to biased outcomes and limited understanding of the decision-making process.

CEM offers a novel approach to address these challenges. By generating contrastive explanations, it provides understandable and actionable insights into the factors that contribute to credit approval decisions. This method helps identify the key features, which it decomposes into Pertinent-Negatives (PN) and Pertinent-Positivess (PP) scores.
To implement this project, historical credit data will be collected and used to train a machine learning model for credit approval. The trained classifier will then be integrated with the CEM model, which will generate contrastive explanations for individual credit applications. These explanations will help both financial institutions and credit applicants comprehend the factors driving the decision outcome and enable them to take appropriate actions.

The project's outcome is expected to yield several benefits, including increased transparency, reduced bias, and enhanced fairness in credit approval decisions. By adopting the CEM methods, financial institutions can make more informed and equitable credit decisions, leading to improved customer satisfaction and better risk management. Furthermore, this project contributes to the growing field of explainable artificial intelligence (XAI) by demonstrating the practical application of CEM in the domain of credit approval.

Keywords: credit approval, IBM Contrastive Explanation Method, interpretability, transparency, explainable artificial intelligence (XAI), machine learning.

Chapter 1

Introduction

1.1 Problem statement

According to IBM researchers, Black-box machine-learning models, such as Deep Neural Networks (DNNs) and large ensembles, have demonstrated remarkable performance in various tasks, despite their inherent opacity to human comprehension. However, the growing incorporation of machine learning models in high-stakes decisions, such as evaluating credit loan applications, necessitates a heightened requirement for these models to be interpretable and explainable [1].

Explainable AI (XAI) is the field of Artificial Intelligence (AI) that attempts to explain the predictions of machine learning models to end-users, often via interpretable algorithms. There are several possible explanations, such as: *data vs. model*, *directly interpretable vs. post hoc explanation*, *local vs. global*, and *static vs. interactive* [2]. In addition, there exist different types of explainers also known as toolkits or models, and often the ideal method depends on the specific consumer demanding the explanation [1].
It is common practice for some texts to refer to XAI as **Interpretable AI** or **Interpretable Machine Learning**. Others may refer to it as **Explainable Machine Learning**. Do note that for the duration of this dissertation, the author shall make use of all these similar terms interchangeably and the reader should assume they mean the same concept.

XAI faces significant challenges that hinder its widespread adoption and efficacy. In the paper *AI Explainability 360: Impact and Design*, *Arya et al, (2021)* [1], the team behind the highly functional AIX 360 AI explainability package, have it that despite the growing volume of publications on XAI, there remains a gap between what the research community is producing and how it can be leveraged by society.

The main reason for this gap as identified by *Arya et al, (2021)* [1] is due to the fact that different people in different settings may require different kinds of explanations. For example, a doctor trying to understand a patient's diagnosis, may benefit from seeing known, similar cases with the patient's diagnosis. While a denied loan applicant will want to understand the reasons for their rejection and what can be done to reverse the decision.

These issues may reflect a lack of standardized methodologies and frameworks or a lack of universally accepted definitions and standards for AI explainability. As highlighted by *Verma et al, (2021)* [3] XAI issues include:

- AI techniques do not share a common vocabulary,
- Lack of quantitative evaluation metrics that can evaluate the explanations of two similar methods such as LIME and xLIME, to ascertain which one best fits a given observation,
- Lack of research on the actionability of ML explanations.

The shortcomings of XAI have not gone unnoticed by regulatory bodies, such as the European Commission [4], The Federal Trade Commission [5], GOV.UK [6] and others [7], [8], who keep evaluating the current regulatory frameworks, to ensure they adequately cover the rapidly-evolving AI landscape. Indeed, the unsatisfactory and largely inconsistent nature of interpretations and explanations regarding the predictions of complex AI models, frequently integrated within modern applications, presents significant safeguarding dilemmas for humans.

For over five decades, financial institutions have predominantly used credit scoring models to determine who receives what in terms of credit and advances. *Arya et al, (2021)* [1] have it that the ideal explanation method depends on the specific consumer demanding the explanation []... This is perhaps most applicable to the financial industry. As it is often customary for large financial institutions to set up specialized teams tasked with the responsibility of upholding transparency and reliability in the implementation of their models. They also propose that the most crucial inquiries confronting major financial institutions as regarding consumer finance are perhaps the following four:

1. Generally, what features are the most important for decisions made by the model?
2. What features drove a decision for a certain input?
3. What features could be minimally changed to alter the decision for an input?
4. Do similar inputs produce the same decision?

These questions offer practical instances that exemplify the financial enterprise's need for an explainability technique, surpassing the general notion that a model should possess the capability to explain its decisions. In handling consumer finance queries, Q1 above sheds more light on the global features of the data that impact the model's decision. It is the kind of explanation that a Data Scientist who evaluates the machine learning model before deployment would be interested in. While Q2 and Q3 are local feature-based explainability questions, and together they can provide a customer with substantive explanations as to what features were most informative and adjustments that may be necessary for future improvements, and in the same vein, Q4 is tailored for the deciding loan officer [1]. In this dissertation, the author shall attempt to improve the model's explanation for Q3 above, which is the explanation given to credit loan customers, especially those whose loans have been rejected. Q3 is an example of a question deserving Local post-hoc explanations. The author shall approach this problem via the **Contrastive Explanation Method (CEM)** [9] and interactive visualization software.

CEM is a novel machine learning explainability method that approaches explanations from the predictions of complex Neural Networks in a minimalistic framework based on contrastive learning. It was chosen for this dissertation by the author, because of its uniqueness and simplicity. Consider a given example $(\mathbf{x_0}, \mathbf{t_0})$ where $\mathbf{x_0}$ is an input and $\mathbf{t_0}$ is the predicted class such that $\mathbf{t_0} \in K\{\mathbf{t_0}, \mathbf{t_1}...\mathbf{t_n}\}$ classes, CEM can readily improve this explanation by decomposing $(\mathbf{x_0}, \mathbf{t_0})$ into Pertinent Negatives (PN) and Pertinent Positives (PP) which combined can simultaneously inform what features and values are *minimally but critically absent* to belong to the definitive class $\mathbf{t_0}$ and *minimally sufficient* without which the predicted class would differ $K\{\mathbf{t_1}...\mathbf{t_n}\}$. Furthermore, CEM goes behind the typical **Black Box Explainers(BBE)** such as the popular LIME and SHAP [10], [11] techniques which are model-agnostic techniques that dwell on the $predict()$ function of a model. In fact, CEM is technically a **White Box Explainer (WBE)**, given that it goes within the trained model and requires access to specific parameters [1]. Furthermore, the author shall utilize the Google What-If-Tool (WIT) for the interactive visualization component of this dissertation. WIT is a dynamic software application specialized in the interactive visualization of complex machine-learning models [12].

1.2 Motivation

By attempting to enhance credit approval decisions for customers, the author's objective is to improve the overall credit processing experience by utilizing explainable artificial intelligence (XAI) and interactive visualization models. Especially for people of colour, who are often underrepresented in financial matters [13], and seemingly underqualified for credit facilities.

This research proposal is driven by the imperative of achieving equity in AI applications, as there is evidence of discriminatory outcomes disproportionately affecting individuals from racial minority groups. For instance, Forbes reports that "*Chicago lenders were **150%** more likely to reject Black applicants than similar white applicants*" [14], highlighting the challenges faced in establishing AI as an impartial scientific endeavour. Consequently, the author asserts that incorporating XAI is essential to ensure fairness in AI.

Finally, I expect my research to inspire future students including students from underrepresented ethnic groups to build XAI systems of the future, devoid of bias and unfairness. World systems that are truly representative of the rich diversity of cultures and people equally.

1.3 Research aims and focus

Building upon the aforementioned objective, the primary aims and focus of this thesis are as follows:

1. Determine Pertinent-Negatives(PN) feature importance scores for rejected loan applicants,
2. Determine Pertinent-Positives(PP) feature importance scores for accepted loan applicants,
3. Extract the Aggregate Statistics for PNs and PPs for pattern analysis.

Correspondingly, the overriding focus of this dissertation is the application of the Contrastive Explanation Method and interactive visualization software to improve the local post-hoc explanations for credit loan customers given their Pertinent-Negatives and Pertinent-Positives scores.

1.4 Structure of the project

This dissertation is divided into five chapters as follows:

- Chapter 1 This section serves as the introductory chapter of the project, where the inherent challenges of Explainable Artificial Intelligence (XAI) are explored. It establishes the motivation behind the project and outlines its specific aims and focal points. Consequently, this section lays the groundwork for the dissertation, establishing its fundamental basis.

- Chapter 2 The background section attempts to provide an overview of the credit scoring industry about artificial intelligence (AI). Additionally, the author delves into a few essential concepts, including Explainable Artificial Intelligence (XAI), emphasizing the involvement of humans in the decision-making process. Lastly, the study explores the current state of Local Post-Hoc explanations as a preliminary step towards understanding the Contrastive Explanation Method (CEM).
- Chapter 3 In this chapter, a detailed explanation is provided of the methodology employed for developing the Contrastive Explanation Model (CEM) and its associated system. The chapter emphasizes crucial concepts, including the interaction between causal and contrastive explanations, and the significance of causal modelling and contrastive learning.
- In Chapter 4, The author applies CEM to a real-life data set and demonstrates the outcomes via pertinent values analysis. By displaying the knowledge acquired through the implementation of CEM, an in-depth analysis of the findings is conducted, supported by appropriate interactive visualizations.
- Chapter 5 Finally, chapter five encapsulates the primary conclusion of this thesis, while also acknowledging its limitations and suggesting potential directions for future work. By emphasizing the essential measures required to develop a more comprehensive model beyond the scope of this thesis, this chapter provides valuable insights and recommendations for future research endeavours.

Chapter 2

Literature review

2.1 Background

2.1.1 AI overview

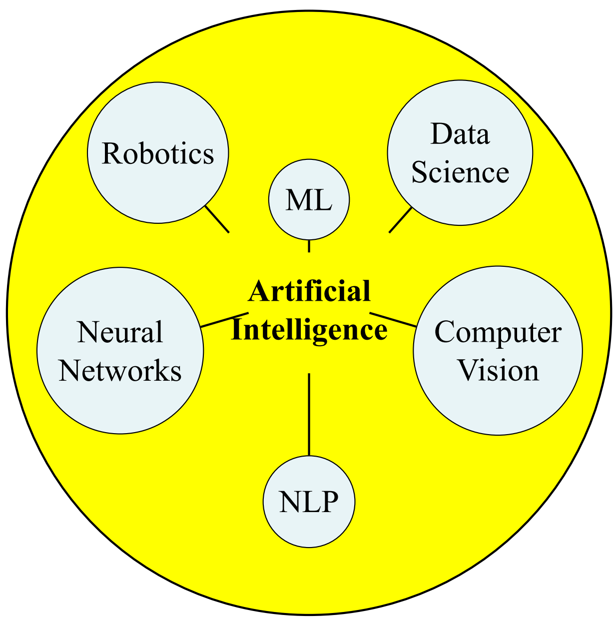


Figure 2.1: *Major Subfields of Artificial Intelligence*

Artificial-Intelligence (AI) needs no introduction. One can barely scan news articles these days, without finding an item or two about AI and its applications. It has become such a buzzword that often, its meaning is lost in translation. Perhaps, to better understand AI, we should consult the most authoritative textbook ever written on the subject [15].

In AIMA (*Artificial Intelligence a Modern Approach*), 4th edition [15], *R. Stuart* and *N. Peter* identified 4 historical approaches to AI.

1. **Thinking Humanly**: The Cognitive Modelling Approach
2. **Thinking Rationally**: The "Laws of Thought" Approach
3. **Acting Humanly**: The Turing Test Approach
4. **Acting Rationally**: The Rational Agent Approach

For The Cognitive Modelling Approach, *Haugeland, (1985)* as quoted in AIMA [15] 4th edition pp. 2, defined Artificial Intelligence as:

"*The exciting new effort to make computers think . . . Machines with minds, in the full and literal sense.*"

This definition is somewhat similar to that of *Winston, (1992)* as quoted in AIMA [15] 4th edition p2, who defined AI under the "Laws of Thought" approach as:

"The study of the computations that make it possible to perceive, reason, and act."

For the 3rd and 4th approaches to AI, *Kurzweil, (1990)* and *Nilsson, (1998)* as quoted in AIMA [15] 4th edition pp. 2, defined Artificial Intelligence as:

"*The art of creating machines that perform functions that require intelligence when performed by people.*"

and,

"*AI . . . is concerned with intelligent behaviour in artefacts.*"

respectively.

Interestingly, the first 2 approaches to AI are concerned with thought processes and reasoning, while the last 2 are concerned with behaviour. Furthermore, *R. Stuart* and *N. Peter* have shown in AIMA [15], that Artificial Intelligence is largely about intelligent agents. Pertinently about how intelligent agents receive percepts from their environments via their receptors, how percepts are processed via the agent functions and programs, and how these agents act on their percepts history via their actuators.

Permit the author to illustrate a simplified version of an intelligent agent. For example, a Large Language Model (LLM), [16] which has been trained on billions of parameters. An example of percepts may be a huge corpus of text data fed to the model, usually in batch or streaming format. Such an agent must have complex functions and programs to process such huge data. Finally, to be useful, it must have actuators often in the form of Application Programming Interfaces (API) and/or Graphical User Interfaces (GUI), that allow for interactions with the world. Such that via prompts, the intelligent agent may change its states as it tries to produce the desired outputs.

It is necessary to state here that Artificial Intelligence and its applications seem to apply to more sectors of modern societies. In the paper *AI Adoption in a Competitive Market* [17], *G. Joshua* posits that in a competitive market, AI adoption is a complement to variable inputs and that in the long run, AI adoption is shown to generally lower prices and raise consumer surplus. This implies that firms that do not adopt AI applications even at the point of predicting and controlling stock levels, risk losing out in the price war.

AI models in various forms and sizes such as Computer Vision models trained via Convolutional Neural Networks(CNN) [18], powering products like self-driving cars or Time-Series models trained via Recurrent Neural Networks(RNN) [19], powering robust financial forecasts or Natural Language Processing models trained via Transformer Architecture [20], powering large scale Text Analytics services such as Sentiment Analysis and Named Entity Recognition, continue to see relatively increased adoption in modern times, albeit with varying results.
Other sectors worth mentioning that have seen an uptick in AI adoption, are digital marketing [21], modern art teaching [22], modern healthcare systems [23], dentistry [24], financial management [25], and vocational education technology [26].

Perhaps, no worthy discussion of Artificial intelligence today can suffice without a mention of Generative AI. According to researchers at IBM,

“ As the name suggests, generative AI generates images, music, speech, code, video or text, while it interprets and manipulates pre-existing data” [27].

IBM researchers also stated that Generative AI is not a new concept. For example, transfer learning, which is an integral step in applying Generative AI, has been in use over the years and simply means fine-tuning a pre-trained model, which has been trained on large amounts of data, to a specific task [27].
So, the game changer for Generative AI was the creation of a new Neural Network architecture called Transformers by the Google Brain Team in 2017 [28]. This has led to the emergence of large foundational models built by combining the Transformer model with unsupervised machine

learning. An example is the ChatGPT Large Language Model(LLM) [29]. Which has progressively achieved outstanding results since its launch in November 2022, surpassing existing benchmarks to handle various data modalities. In fact, the author prompted ChatGPT for the full meaning of its name and the quote below, which is also in the appendix, is the response it gave:

> “ *The term ”GPT” stands for Generative Pre-trained Transformer. Generative refers to the model’s ability to generate text and other content, Pre-trained indicates that the model is trained on a large corpus of data before being fine-tuned for specific tasks, and Transformer refers to the architecture of the neural network used in the model. So, ”GPT” essentially describes the model’s nature as a pre-trained generative text model based on the Transformer architecture.* ”

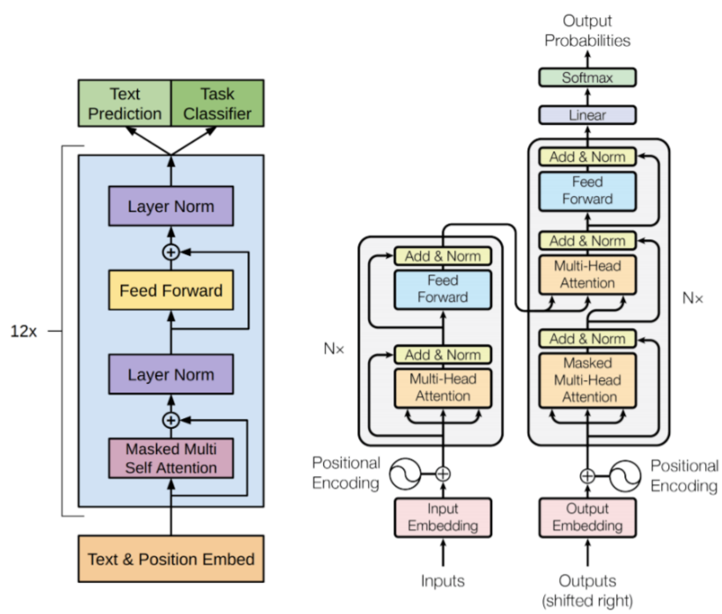


Figure 2.2: *The Transformer Architecture* [30]

2.1.2 Credit approval overview

Despite the ground-breaking achievements in modern-day AI and its applications to the financial sector, it remains common knowledge that financial institutions are constantly targeted by fraudsters and unscrupulous elements, who also deploy sophisticated phishing algorithms, seeking to illegally profit from possible loopholes and security breaches, including data breaches. In the paper *Machine Learning in Finance* [31], *Kumar et al, (2021)* have it that the finance sector encounters a perpetual array of challenges, encompassing credit card fraud, identity theft, network intrusion, money laundering, human trafficking, and illicit firearms transactions. Prevalent crimes such as credit card fraud, identity theft, network intrusion and money laundering have become quite sophisticated and require novel solutions. In addition, novel threats are emerging, such as the proliferation of false information in financial media, which has the potential to distort trading strategies and investment choices.

But perhaps, the biggest risk faced by financial institutions is credit risk [32]. This appears to be a multifaceted problem that a lot of financial institutions are struggling to curb. According to *Mckienzie (2021)* [33], a significant obstacle preventing certain banks from implementing more advanced credit scoring models to manage credit risks is their heavy reliance on subjective evaluations from relationship managers and underwriters.

For over 50 years, financial institutions have predominantly used credit-scoring models to determine who receives what in terms of credit and advances [32]. Scholars and industry experts have devised diverse credit scoring instruments to tackle the distinct decisions encountered at various stages of the credit risk management process. ***Application scoring*** is employed to determine the eligibility of loan applicants, while ***Behavioural scoring*** serves to delineate the characteristics of borrowers who have already obtained a loan, primarily for portfolio management purposes. Lastly, ***Collection scoring*** aids in optimizing collection and recovery policies and strategies *(Paleologo et al., (2010)* as cited in [32].

For the sake of this dissertation, the author shall dwell on ***Application scoring***, which is by far the most common credit-scoring method. Application scoring models play a crucial role in determining the approval or denial of loan applications, serving as the gateway to the lender institution and the broader financial system. Therefore, it is of utmost importance to possess models that accurately quantify the level of risk associated with borrowers and possess a high predictive capability to determine whether the applicant is likely to default on the loan or not. Perhaps we should take a moment to inspect the components of a credit score.

“A credit score is a number that rates your credit risk” [34]. Creditworthiness is typically determined using a credit report, which provides a snapshot of an individual’s financial reliability at a specific moment. According to the US Consumer Financial Protection Bureau (CFPB) [35], credit scores are primarily based on an individual’s credit history and behaviour. Typically ranging from 300 to 850, higher credit scores correspond to more favourable interest rates. The table below shows common factors that may affect one’s credit score.

Table 2.1: *Common Factors Affecting Credit Score*

Factor	**Description**
Payment History	History of on-time payments and delinquencies
Credit Utilization	Percentage of available credit being used
Credit History Length	Length of credit accounts' history
Credit Mix	Variety of credit types (e.g., loans, credit cards)
New Credit	Recent credit inquiries and new accounts opened
Credit Account Balances	Amounts owed on credit accounts

Historically, Credit Bureaus employ credit scoring models, which utilize mathematical formulas, to calculate credit scores. These models primarily assess the likelihood of default, often determined via the Logistic Regression algorithm [36], denoted by the *Sigmoid* activation function $\sigma(x) = \frac{1}{1+e^{-x}}$. This function has the remarkable property of squashing numbers into the $(0 - 1)$ range. Thereby representing the scores as probabilities, so with a set threshold $t \approx\leq 0.5$, along the *Sigmoid* curve, the algorithm can simply compute the probability of default per customer x, where x is a vector of features values, features such as the factors in Table 2.1 above.

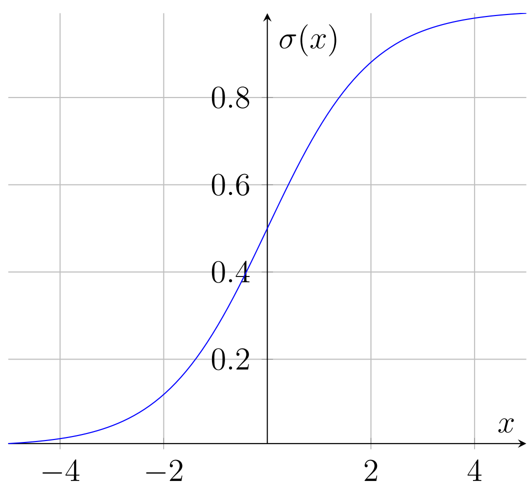


Figure 2.3: *Sigmoid Curve*

While this approach is fair enough, Logistic Regression is a parametric model, that tries to learn a function that approximates $P(Y|x)$, where P is the probability, Y is the target column vector, and x is an input row vector. This is based on the assumption that $P(Y|x)$ can be approximated as a Sigmoid function applied to a linear combination of feature values. These inherent presumptions of the intricate data patterns made by many parametric algorithms seldom scale as data complexities increase. Often, non-parametric algorithms such as DNNs demonstrate superior generalization capabilities when faced with complex, non-linear data patterns. Generally, as the complexity of a model increases, its capacity to approximate complex functions or patterns within the data also improves, albeit at the expense of diminished interpretability and transparency.

As earlier mentioned, application scoring is employed to determine the eligibility of loan applicants. In the paper titled *On the dynamics of credit history and social interaction features, and their impact on creditworthiness assessment performance*, *Muñoz-Cancino et al (2022)* [32] further identified some noteworthy factors that contemporary application scoring algorithms integrate to infer their decisions. These factors include the applicant's demographic details, loan repayment track record, borrower historical data, credit bureau information, and data gathered from the application form itself. Thus, the credit risk management decision involved in application scoring is to selectively approve loans for borrowers with high repayment capacity while avoiding credit approval for those with high default risk. Application scoring facilitates proactive measures, such as alleviating financial burdens for borrowers facing payment difficulties, based on their likelihood of default [32].
Figure 2.4 depicts a very simplified flow chart of activities leading up to a credit approval decision. It is important to connect these to the 4 questions asked earlier by *Arya et al, (2021)* [1] in section 1.1. Which the author has graciously repeated below, for the sake of ease, and given that these questions are perhaps most structural to any credit explanation system.

1. Generally, what features are the most important for decisions made by the model?
2. What features drove a decision for a certain input?
3. What features could be minimally changed to alter the decision for an input?
4. Do similar inputs produce the same decision?

By providing accurate responses to these four questions, the decision-making process can be enhanced, effectively fulfilling the relevant criteria of the three primary stakeholders in any lending institution:

- **The Data Scientist:** For Q1 above, in evaluating machine learning models for deployment, Data Scientists must understand the model's behavior holistically, not just in specific instances. This is crucial in regulated industries like banking, where higher explainability standards apply. The model may need to be presented to:

    1. Technical and business managers for review,

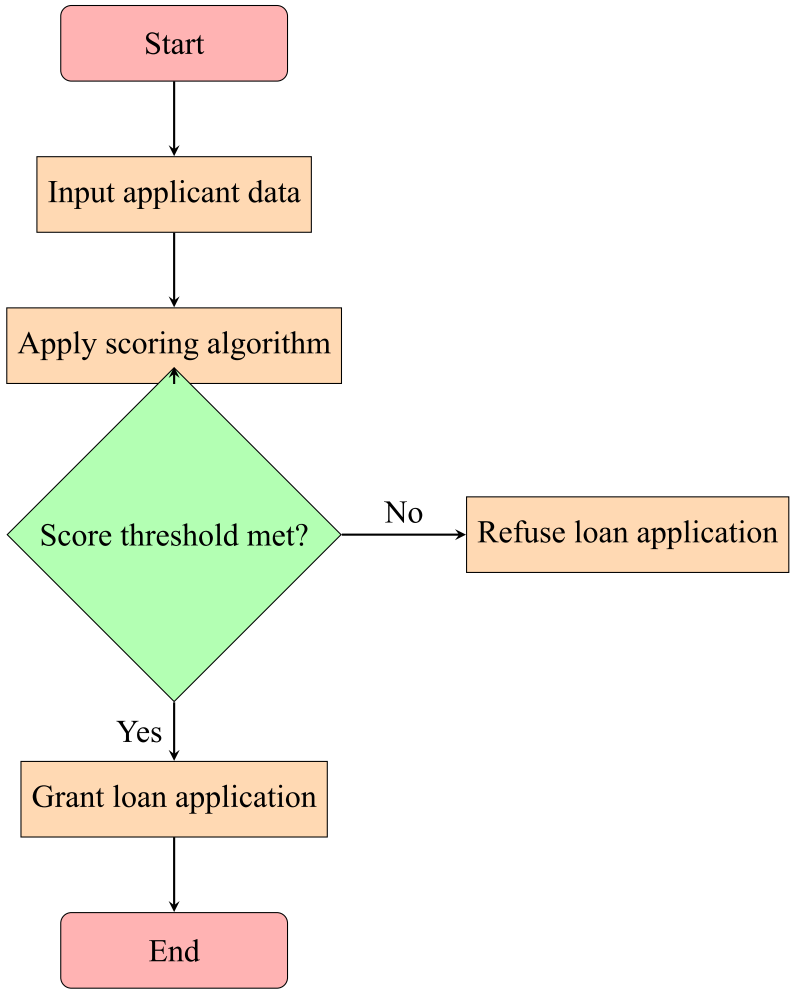


Figure 2.4: *Simplified Loan Application Scoring Decision Flowchart*

2. Lending experts for comparison, or
3. regulators for compliance checks.

Additionally, a global perspective can reveal issues with the model such as over-fitting and poor generalization prior to implementation.

- **The Deciding Loan Officer:** Actively interested in Q4 and relatively interested in Q2, the Loan Officer needs to understand the decision of an applicant's application being accepted or rejected in the context of other similar applications. Similarity measures are most important to this stakeholder. It helps them to justify loan decisions and creates readily available data for domain experts to critique as feedback for improving the model's decision.
- **The Customer:** Loan or credit applicants desire insight into why they are ineligible for credit and what changes in their application would qualify them. This refers to Q3 above.

Conversely, if approved, they seek to understand the factors that influenced the decision. Contrastive explanations provide applicants with information on the minimal profile adjustments that could have altered the AI model's decision from rejection to acceptance or vice versa (pertinent negatives). For instance, reducing the defaulted loan repayment days to a specific value may have led to application acceptance, *ceteris paribus*.

This dissertation focuses on the improvement of explanations provided to customers, after loan decisions. Explanations such as these are largely contrastive and reflect the customer's desire to be informed of what they need to do to get a favourable response if their loans were denied, or if approved, what minimum criteria could have yet maintained it. Contrastive explanations are discussed in-depth in the next chapter. The author bases his research and approach on the novel Contrastive Explanation Method (CEM) developed by IBM researchers in 2018.

2.2 Key concepts

2.2.1 Explainable Artificial Intelligence (XAI)

In the *AI explainability white paper, (2019)* [37], Google Cloud Engineers have it that the quest for more accurate AI has led to complex non-linear models, fueled by abundant compute resources and larger datasets. This evolution spans from handcrafted rules to linear models, decision trees, ensembles, deep models, and recently, the emergence of meta-learning or model-generating models. As model complexities increase from basic parametric structures like multiple linear regression to large, complex architectures such as Deep Neural Networks (DNNs), commonly referred to as Black-box models, it becomes evident that model explainability, interpretability, and transparency are likely to be compromised.

This is the context where Explainable Artificial Intelligence (XAI) may demonstrate its relevance or offer some relief, albeit somewhat far-fetched given the complex needs and varying types and formats of explanations required by humans [38]. As AI becomes increasingly integrated into contemporary applications and strategic decision-making processes, such as credit lending, the precise definition of XAI could be as varied as that of AI. Gratefully, the author would not attempt to cite or discuss any more definitions of XAI from the numerous scientific papers in circulation. Rather, the author prefers to dwell on the central or general concepts, which include:

1. The fundamental notion of XAI a person with some basic education might expect.

2. The potential effect of Humans-In-The-Loop, in the realms of XAI.

At its barest minimum, Explainable AI (XAI) should be as it sounds. One could argue that an average person on hearing these words would expect AI to be *explainable*... Moreover, depending on an individual's expectations and understanding of AI, the perception of XAI may vary. But considering the prevalent excitement and almost mystical aura surrounding AI, particularly in news and social media, the author holds the view that the general public anticipates XAI to revolve around processes, structures, tools and perhaps guidelines or regulations that can help demystify how AI influences decision-making. And perhaps on a rational level, how AI arrives at its predictions of future outcomes given only past and in seldom circumstances, present data. Consequently, irrespective of our differing definitions or expectations of AI, the author makes a bold claim that in the realms of XAI, everyone expects these explanations to be discernible, comprehensible, largely rational, and meaningful to humans.
Truly, the world is warming up to the *Rise of The Machines*..., And it is not just relegated to high-tech or computationally intensive sectors like Banking, Finance, and Engineering. But smart devices, expert system machines, intelligent agents, machine-learning models and DNNs are gradually powering the logic and outputs of several sectors and in a few cases, economies. Let us inspect one prominent regulatory framework for safeguarding trustworthy AI: The European Commission Ethics Guidelines for Trustworthy AI [39]

Table 2.2: *European Commission's Key Requirements for Trustworthy AI Systems*

Key	Summary
Human agency	AI should support human decisions and autonomy, not replace them.
Technical robustness and safety	AI systems should be secure, reliable, and resilient to errors.
Privacy and data governance	Personal data used in AI should be protected and handled ethically.
Transparency	AI systems should be explainable, providing clear and understandable results.
Diversity, non-discrimination, and fairness	AI should be unbiased, inclusive, and prevent discriminatory outcomes.
Societal and environmental well-being	AI should benefit society and respect environmental sustainability.
Accountability	AI systems and their developers should be accountable for their actions.

Table 2.2 shows a summary of the 7 key requirements for trustworthy AI as delineated by the European Commission. Fairness, transparency and accountability are prerequisites for building trustworthy AI systems, as such, the summaries for *Transparency* & *Accountability*, directly apply to explainability.

Recall from the *AI overview* section of this chapter, that *R. Stuart* and *N. Peter* promulgated the narrative that AI is about intelligent agents. In the same text, they identified the Rational Agent approach as the most ideal, for two main reasons ([15],*4th edition pp. 4*).

1. It is more general than the other approaches. For example, the Laws of Thought approach focuses on correct inference, but in the real world, correct inferences are but just one of several possible mechanisms for achieving rationality

2. In the spectrum of scientific advancement, rationality sets a well-defined and universally applicable standard, allowing for the creation of agent designs that can reliably achieve it, backed by mathematical proof.

The rational agent approach seems to underscore Aristotle's opinion from the text *De Motu Animalium, (cited in AIMA 3rd Edition, pp 7, [15])* [15], in which he argued that ***Actions are justified by a logical connection between goals and knowledge of the action's outcome***... Thus if the goal of a student at The University of Wolverhampton were to be in class by 10 am tomorrow, to take a test and they choose to take the bus that departs by 8:30 am, at a fare of 4 pounds, to arrive by 9:50 am tomorrow, one may infer from Aristotle's statement above that since there is a logical connection between taking the bus, with the likely outcome of arriving 10 minutes before time to achieve the goal, despite spending 80 minutes travelling and 4 pounds fare, then such actions are justified. However, in the *Nicomachean Ethics (Book III. 3, 1112b)*, Aristotle provided a more comprehensive account by explicating an algorithm, which entailed the sequential progression from an initial state to a desired goal state. Additionally, he introduced the concept of rationality, specifically emphasizing the pursuit of optimal problem-solving. This significant contribution by Aristotle is referenced in the *AIMA 3rd edition (pp. 7, [12])* [15]. Aristotle's notable statement within this context is as follows:

> " *We deliberate not about ends, but about means. For a doctor does not deliberate whether he shall heal, nor an orator whether he shall persuade, . . . They assume the end and consider how and by what means it is attained, and if it seems easily and best produced thereby; while if it is achieved by one means only they consider how it will be achieved by this and by what means this will be achieved, till they come to the first cause, . . . And what is last in the order of*

analysis seems to be first in the order of becoming. And if we come on an impossibility, we give up the search, e.g., if we need money and this cannot be got; but if a thing appears possible we try to do it. ”

So in this scenario, the student may consider taking a cab and spending only 20 minutes of travel time, for 9 pounds, so they could arrive earlier to brush up for about an hour in the library, before taking the test, rather than sitting in the bus for an extra hour. Although the cab may be more expensive, it may be the rational option and thus justifiable, if the student needs the extra time to brush up for the test, given that passing the test could earn the student a scholarship. Therefore, the author argues that the justification of the action given the goal must be such that the action is the best possible one given all the information and perceptions available to the student, just as Aristotle argued that “*For a doctor does not deliberate whether he shall heal, nor an orator whether he shall persuade, . . . They assume the end and consider how and by what means it is attained, and if it seems easily and best produced thereby;*”, *AIMA 3rd edition (pp. 7, [12])* [15]. Precisely summing up the Rational Agent Approach to AI.

Intriguingly, given the definition of a rational agent by *R. Stuart* and *N. Peter* as one who acts so as to achieve the best outcome or when there is uncertainty, the best-expected outcome. Humans frequently fail to meet this criterion of rationality through our actions, as demonstrated by the multitude of subpar decisions we consistently make. This occurs despite the wealth of historical data and personal experiences at our disposal, which could serve as our percept sequence within the framework of agent terminology. This portends greater challenges to XAI, given that one explanation is not enough, as seen in [38], different types of users demand different styles of explanations. How therein can we create uniform explanations for similar users, since we struggle to standardise intelligence and rationality and even when we do, our actions often fall below these standards? We may suggest that the domain experts should decide on the structure and the appropriate responses for respective users per domain, and although this may seem practicable, it is susceptible to human errors and biases.

Referring to the *AI explainability white paper, (2019)* [37], Google Engineers described a condition known as the **human analysis pitfalls**. By this, they disclosed that although XAI revolves around empowering the human in the loop, the field of cognitive psychology, states that an essential insight regarding human reasoning is that rationality is not always employed consistently. Instead, individuals frequently rely on heuristic reasoning, which is a fast, low-effort process that enables faster decision-making, using our subconscious, cognitive shortcuts. For example, while trying to identify someone’s face, people tend to utilize heuristic approaches, including mental clustering and comparison, to faces they are familiar with. As a result, it is not uncommon for friends and acquaintances to bear resemblances to popular celebrities. This can be attributed to the brain's tendency to generate comparisons starting from familiar reference points. This once again underscores the previous point that we don't often reason rationally, and by extension, neither do we often act rationally and probably unintelligently. The paper further identifies the following challenges to XAI due to how humans often think (heuristically or analytically).

- Models and humans may approach prediction tasks differently, yet both can be correct. The presence of highly correlated/redundant features in data may cause the model to perform accurately using different feature combinations, which may result in predictions different from human expectations, this by no means indicates a model error.

- Related to the above point, models aim to distinguish among specific labels, often without considering the full range of features utilized by humans to discriminate objects in the real world. Users may overlook this and anticipate explanations encompassing a broader set of features than those employed by the model. For instance, features localized to select regions indicating a lung cancer diagnosis in an X-ray may suffice for the model's prediction.

- Finally, There is a possibility of human perception misjudging the model's correctness. A human may consider the model incorrect even when it is accurate or perceive it as correct due to confirmation bias. In this situation, they may also doubt the feature importance values within the models' prediction.

While the need to address and interpret the predictions of Black-box models seems to be the focus of several research in XAI, the author felt the need to highlight the Human-In-The-Loop challenges that so often result in bias or stem from bias. This invariably infers that in certain circumstances, the challenge may not be XAI, but the reasoning of the human user in the loop.

C. Kozyrkov, the Chief Decision Scientist at Google, presented a somewhat pessimistic viewpoint on the challenges surrounding the relationship between humans and XAI in the article titled *Explainable AI won't deliver. Here's why [43]*. Herein she discussed that complexity is the reason for all of it. This is because some tasks are so complicated that humans cannot automate them by giving explicit instructions, and in such circumstances, the focus should largely be on explaining with examples, rather than trying to figure out the specific instructions within the model leading to a decision. That is the job of the AI algorithm, not the human. *C. Kozyrkov* promotes the principle of testing as a better basis for trust than an attempt to simplify the decision-making mechanisms of complex DNNs into human-discernible explanations.

Table 2.3: *Popular Explainable AI Techniques*

Technique	Summary
LIME [11]	Local Interpretable Model-agnostic Explanations (LIME) provides explanations for any black-box model by approximating its behaviour locally.
SHAP [10]	SHapley Additive exPlanations (SHAP) assigns each feature an importance value for a model's prediction, considering all possible feature combinations.
Grad-CAM [40]	Gradient-weighted Class Activation Mapping (Grad-CAM) generates heatmaps to visualize important regions in an input image that contribute to the model's decision.
Rule-based Methods [41]	Rule-based methods generate human-readable rules or decision trees to explain the model's behaviour based on its learned decision boundaries.
Feature Importance [42]	Feature importance methods rank the importance of input features based on their contribution to the model's predictions, aiding in understanding feature relevance.

" If you refuse to trust decision-making to something whose process you don't understand, then you should fire all your human workers, because no one knows how the brain (with its hundred billion neurons!) makes decisions, *C. Kozyrkov, (2021), [43]*. "

Having critically examined the position of *C. Kozyrkov, (2021)* and the challenges discussed above, the author acknowledges the significance of robust testing for AI models but emphasizes the need to intensify research in Explainable Artificial Intelligence (XAI) to foster trust. Trust stems from rigorous testing and an improved understanding of AI decision-making processes as it becomes integrated into everyday life.

To conclude this discourse on XAI, it is pertinent to mention the notable contributions made by *E. Austin*, from the *IBM AI Explainability Master Class [2]*. The author takes pride in being a recipient of the distinguished *AI Explainability* badge and an active participant in this program. *E. Austin* and team approach explainability as a workflow:
The XAI workflow plays a pivotal role in mitigating irregularities and errors that may impact the accuracy of model predictions. It encompasses the entire AI developmental project, fostering a culture of transparency and accountability among Data Scientists. This involves sharing the rationales behind decision-making, ranging from mundane tasks like proper documentation and naming conventions to logical justifications for selecting specific model architectures. The author asserts that XAI should be embraced as an iterative workflow with the overarching aim of

achieving human-like explainability in AI. It is incumbent upon each individual to contribute by producing well-documented scripts and actively participating to curb the challenges of XAI.

2.2.2 Local Post-hoc Explanations

Before zooming in on local post-hoc explanations, let us take a few minutes to get acquainted with some important concepts. Generally, machine-learning explanations or interpretability can be classified into **Global** and **Local** explanations. In the paper *Techniques for interpreting machine learning [44]*, *Du et al (2019)* discuss that:

- Global explanations relate to the ability of users to acquire a comprehensive understanding of a complex model's operations by scrutinizing its structures and parameters. They are particularly valuable for debugging purposes or gaining insights into the features that significantly influence the model's predictions (feature attrition).
- While Local explanations focus on analyzing individual predictions made by the model, aiming to discern the reasons behind the specific decisions it produces

Global interpretability has the potential to enhance transparency by revealing the internal mechanisms of machine learning models. In parallel, local interpretability aids in revealing the causal relationships between individual inputs and their respective model predictions. These two aspects contribute to user trust in both the model and its predictions, aligning with the European Commission's objective of establishing trustworthiness in AI [39].

Du et al (2019) further identified two distinct techniques for interpretable machine learning: **intrinsic** interpretability and **post-hoc** interpretability. Intrinsic interpretability refers to the construction of self-explanatory models that integrate interpretability into their inherent structures. This category encompasses a range of models, including decision trees, rule-based models, linear models, and attention models.
Parametric models like simple linear regression or the Generalised Linear Models(GLM) often possess intrinsic explainability. Frankly, the main difference between intrinsic and post-hoc interpretability as you may have guessed, is the trade-off between performance such as accuracy and interpretability fidelity.

Post-hoc explainability involves developing a second model for explaining an existing model. It could also be global or local. As expected, global post-hoc explainability aims to decipher and retain pre-trained models' acquired knowledge and parameters, providing humans with an

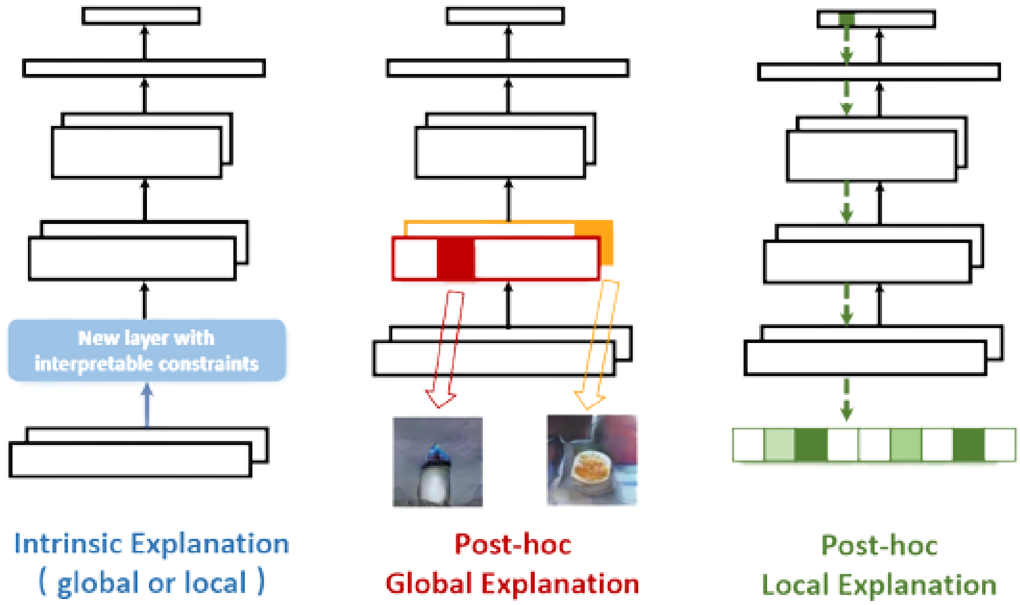


Figure 2.5: *Cross section of global and local XAI techniques* [44]

intuitive understanding of their learned representations. While local explanations aim to identify the feature attrition values of each feature towards a specific model prediction [44]. The concept of post-hoc explainability is akin to that of surrogacy. In fact, The interpretable model can be seen as a local surrogate for the original black-box model [45].

CEM is a local Post-hoc explanation method, which needs to introspectively access a given complex model in order to approximate the model parameters needed for its explanation. In this regard, it is classified as a White Box Explainer (WBE). This is different from Black Box Explainers (BBE) such as LIME [11] and SHAP [10] which are model-agnostic explainers that only require the model, a $predict()$ function and perhaps a set of semantically meaningful features $\mathcal{X}$ [1].

Chapter 3

Contrastive Explanation Method

3.1 Causal explanations

"*According to a causal model of explanation, we explain phenomena by giving their causes or, where the phenomena are themselves causal regularities, we explain them by giving a mechanism linking cause and effect*", *Peter Lipton (2010)*.

The above quote was taken from the paper *Contrastive Explanation* by *Peter Lipton* [46]. In this paper, *L. Peter* discusses contrastive explanations against the backdrop of causal explanations. The general idea is quite intuitive. For example, if we explain that binge drinking causes a hangover, we do not need to give the cause of this causal connection, rather, we simply give the causal mechanism that makes it happen. In other words, we do not need to explain why binge drinking could cause a hangover, but simply stating the causal mechanism, in this case, binge drinking, is sufficient an explanation. This approach to explanations is both natural and plausible. *L. Peter* further argues that the causal model also avoids many of the objections to other accounts of explanations such as the views that:

- To explain a phenomenon is to give a reason to believe the phenomenon occurred.
- To explain a phenomenon is to give a Deductive-Nomological (DN) [47] argument for the phenomenon

Unlike the belief account above, a causal model establishes a conspicuous differentiation between comprehending the underlying reasons behind a phenomenon and simply acknowledging its existence. Citing the previous example, a causal explanation such as "Binge drinking causes a Hangover" is by itself sufficient without a thorough understanding of the underlying cause of a hangover. Thus, a causal model accomplishes this distinction in a manner that eliminates the

enigma surrounding understanding the phenomenon itself, this renders the explanation more objective. It is essential to note that in this regard, understanding is not an extraordinary form of knowledge, but rather an augmentation of knowledge itself.
And in contrast to the Deductive-Nomological case, a causal model in itself does not necessitate the establishment of a law to identify a cause. Intriguingly, one does not require knowledge of any law to possess a rational basis for believing that a cause is indeed a cause.

L. Peter further buttresses the above point by stating that Mathematicians and Philosophers give explanations, but mathematical explanations are never causal, while philosophical explanations seldom are [46]. While the causal explanation model by itself seems plausible and the author largely agrees with the facts and opinions expressed by *L. Peter*, the connection between causal explanations and contrastive explanations so far remains blurry. The reader may wonder wherein the connection lies. Without searching far and wide it is pertinent to comprehend that according to a causal model of explanation, we explain phenomena by giving their causes [46].
The Oxford Dictionary online defines a phenomenon as:

"*A fact or an event in nature or society, especially one that is not fully understood*"

Thus, a phenomenon could be a Black-box model, a Deep Neural Network (DNN) architecture or a Large Language Model (LLM), comprising billions of parameters, whose inner workings are not fully understood. Remarkably, *L. Peter* makes a fine attempt at highlighting the connection between causal and contrastive models using the ***Fact & Foil*** principle.

To understand the ***Fact & Foil*** principle, think of the causal model of explanation for a fire... Both the spark and the presence of oxygen contribute to the causal history that culminated in the occurrence of the fire. However, it is important to recognize that only one of these factors (The spark), tends to explain the fire. So, the logical question that follows is "*What differentiates one piece of information regarding the causal history of an event as explanatory while another piece is not?*" Well, in the paper *Contrastive Explanations*, pp. 3, [46], *L. Peter* succinctly asserts that the cause that explains an event depends on the interest of the enquirer.
This might seem somewhat vague or lacking scientific merits, but before we dispel this notion, it is important to consider the position of *(Hempel, 1965, pp. 421-3)* as cited in [46], which implies that we do not explain events, but only aspects of events. For example, think of the eclipse, our explanation of the eclipse focuses solely on the duration, partiality, and visibility of the eclipse. We may focus on why it was not visible from a certain part of the world, this reduces the number of causal factors we need to consider for the eclipse or any particular phenomenon, based on our interests.

Thus, in the paper *Contrastive Explanations*[46], *L. Peter* asserts that within the context of seeking explanations, individuals do not merely inquire about the causes behind the occurrence of event **P**; instead, they probe into the reasons for event **P** taking place instead of an alternative event **Q**, which may not be explicitly stated. Event **P**, referred to as the "*fact*" or the "*target event*", is the factual occurrence, while event **Q**, known as the "*foil*", represents the contrasting event that did not transpire, which often is the explanation that individuals seek.

Fact	Foil
Mary won the Award	Why did Mary win the Award? (*implicit*) Why did Mary win the Award, but not John? (*Explicit*)

Table 3.1: *Fact & Foil: Literary Award*

Table 3.1 above shows an example of implicit and explicit foils for a fact that occurred after *Mary* and *John* competed for the Writer's Award. Chances are anyone aware that the writing competition was between *Mary* and *John*, yet asks the implicit foil question is likely interested in the explanation for the explicit foil. Likely, they may want to know what *Mary* did better than *John* especially if they thought *John* was as good a writer as *Mary*. Even if they assumed *Mary* was better at writing than *John*, they may well be interested in knowing what *John* needs to improve, to be in a better shape to win in the future. This is in tandem with human communication and nature. And hopefully, the *Fact* & *Foil* principle has set a worthy background to enhance the reader's understanding of contrastive explanations.

3.1.1 Modelling causal explanations

"*Causal models are mathematical models representing causal relationships within an individual system or population [48].*"

The above definition is taken from the article *Causal Models* by *C. Hitchcock, (2018)* [48]. Causal models can be used to improve causal explanations. Basically, these are mathematical tools for the facilitation of inferences about causal relationships from statistical data. In designing causal models, *C. Hitchcock* identified three mathematical tools:

1. Variables, Logic, and Language
2. Probability

3. Graphs

In this section, The author shall produce a graph depicting an example of a Structural Equations Model (SEM)[48], which is simply a type of causal model. SEM makes use of a set of variables, and a set of equations, describing how each variable depends upon its immediate causal predecessors.

To model a SEM graph, let us consider a Masters's Student's journey to the University of Wolverhampton. We can describe the steps taken with the following variables:

- Apply: {0=No, 1=Yes}
- Admission: {0=No, 1=Prov, 2=Full}
- Register: {0=No, 1=Pending, 2=Full}
- Modules: {1=Start, 2=Complete}
- Internship: {0=No, 1=Yes}
- Project: {0=No, 1=Incomplete, 2=Done}
- Viva: {0=No, 1=Yes}
- Graduate: {0=No, 1=Pass, 2=Merit, 3=Distinction}

So, we have the general activities that new students may likely undertake as they enrol at the University of Wolverhampton. These activities have been abstracted into variables. First, a student would need to apply, if not he/she can not be admitted. Next, admissions may take one of <None>, <Provisional> or <Full> statuses, depending on the student's peculiar circumstance. Such a student would need to register for the right academic session and modules. Next, there is an optional internship as part of school activities, that the student may or may not partake in. After all the modules are cleared successfully, the student has to pass their project and viva presentation, before they could graduate.

For example, $\text{Graduate} = 0$ means the student did not graduate, this could be that they dropped out or were not admitted fully or any other probable cause. $\text{Internship} = 1$ means the student undertook an internship, and so on, for the remaining variables.

Then some of the equations could be:

- $\text{Registered} = \text{Apply} \times \text{Admission}$

- $\text{Graduated} = \text{Project} \times \text{Viva}$
- $\text{Project\_Done} = \text{Register} \times \text{Modules} + \text{Internship}$
- $\text{for\_Internship} = \text{Admission} \geq 2$

So, we can see the following *effect variables* and their respective *cause variables*. This is represented as a variable/value statement. With the effect variables on the left of the = sign and the cause variables on the right, this is the default writing convention for this relationship. The effect variable Registered takes the values of Apply and Admission as cause variables. While Graduated does the same for Project and Viva. for_Internship takes Admission as its only cause variable, in a nice boolean equation that evaluates to **True** or **False** depending on whether the Admission value is ≥ 2.
Technically, a SEM is acyclic if the variable(s) that appear on the left side of the = sign do not appear on the right too, in the same equation. Therefore, this example is acyclic and the author shall depict the corresponding Directed Acyclic Graph (DAG) in Fig 3.1 below;

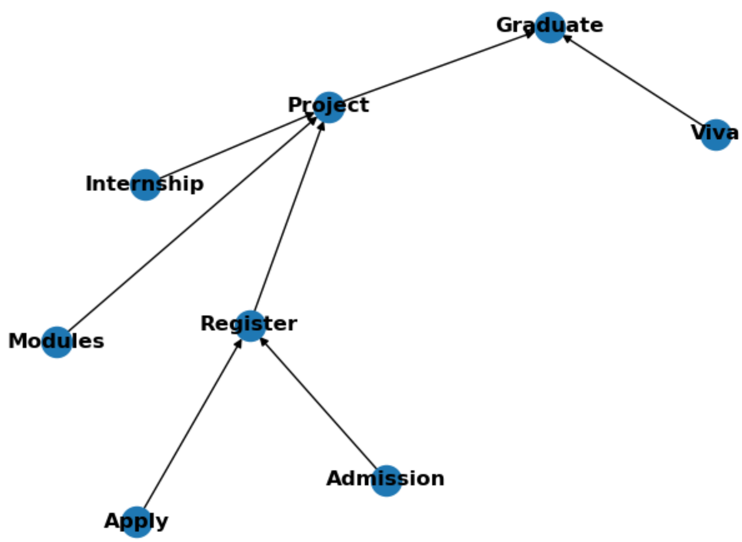


Figure 3.1: *The Causal Model DAG*

In particular, DAG provides qualitative insights into the inter-dependencies among variables, highlighting their mutual dependencies. However, it does not provide any information regarding the specific functional relationship or form of such dependencies. In other words, while we can clearly see from Fig 3.1 that the effect Register depends on the causes $\{\text{Apply}, \text{Admission}\}$, we cannot explicitly tell the values or the weights of the causal relationship between these three nodes. This limitation can be effectively addressed in Graph theory by assigning weights to the connections between nodes and adjusting the visual representation to display these weights. The primary purpose of this exercise was to provide the reader with concrete causal explanations within the mathematical framework. Furthermore, *C. Hitchcock, (2018)* [48], demonstrates the modelling of counterfactual explanations via probability. This seems reasonable as probability axioms can be applied to the measurement of uncertainties and the likelihood of events occurring. for example, given a certain fact **P**, what is the conditional probability that a specific or related foil **Q** would occur? This is a non-trivial exercise as observed by *C. Hitchcock*.

$$P(\mathbf{Q}|\mathbf{P}) = \frac{P(\mathbf{Q} \cap \mathbf{P})}{P(\mathbf{P})}$$

3.2 Contrastive explanations

It is the author's opinion that a good understanding of the causal explanation model discussed above would likely enable the reader to better comprehend the contrastive explanation method. Interestingly, as argued by *L. Peter*, "*an explanation is 'interest relative', and we can analyse some of this relativity with a contrastive analysis of the phenomenon to be explained.*". Thus, What gets explained is not simply "*Why this?*", but "*Why this rather than that?*" *(Garfinkel, 1981, pp. 28—41)* as cited in [46]. In so doing, the contrastive analysis of explanations seems very natural. As earlier stated, *L. Peter* adopts the stance that a contrastive phenomenon consists of a ***fact*** and a ***foil***, and the same fact may have several different foils. We can liken a fact to the outcome of an event. For example "*Mary emerged as the winner of the writing competition*" is a fact. A foil in this situation could be "*Why did Mary win the writing competition and not John?*", this makes the contrastive explanation model natural and easy to grasp. People often present our why questions in a contrasting manner, and it is not difficult to find instances where individuals select diverse foils, thereby demanding distinct explanations.

It is easy to see how contrastive explanation models are highly suited for customer explanations in credit approval queries. As we observed earlier a Black-box model such as an LLM is a phenomenon. This means its internal workings are not fully understood nor readily explainable. Such complex models are often deployed by financial organisations for credit application scoring purposes. The causal explanation model cited by *L.Peter* as a backdrop for contrastive explanations sheds light on the notion that we can explain a phenomenon by giving its cause or where the phenomenon is a causal regularity, we explain it by giving a mechanism linking cause and effect [46]. Thus, we absolutely may not need to understand how every single parameter within an LLM is randomly tweaked or adjusted as the algorithm trains, for that algorithm to be explainable. What we should rather concentrate on is the explanation of the mechanism linking the cause and effect of the LLM, using contrastive explanations suited for different users. Thus, the cause and effect for a credit loan customer could simply be the application of the LLM to that customer data and the effect of its prediction for that customer.

It is relevant to note that in certain scholarly works, **Q** is occasionally denoted as the "counterfactual", which can lead to confusion when analyzing causality, as counterfactuals are dependent on the cause **C** of event **P**. These counterfactual scenarios consider hypothetical instances in which cause **C** did not occur. Conversely, foils pertain to the outcome itself and involve hypothetical alternative outcomes for event **P** [49]. However, for the sake of simplicity, the author shall denote foil and counterfactuals as having the same meaning, to represent the contrasting event that did not occur, which often is the explanation that we seek.

Recall that **P** could be the outcome of an event, but **Q** can be likened to a contrast or a counterfactual case. This counterfactual case could either be compatible or incompatible with the fact. Compatibility represents the likelihood of occurrence between **P** and **Q**. If **P** and **Q** form a set of 2 elements and either is likely to happen, but only **P** occurs, then it is referred to as a compatible relationship as **P** is a subset of the set containing **P** and **Q**.

$$P \subset \{P, Q\}$$

While an incompatible relationship represents a situation where if one occurs, the other cannot occur, denoted by a mutually exclusive set.

$$\{P \cap Q\} = \{\}$$

For example, if *Mary* and *John* were competing for the Writer's Award and *Mary* won. Table 3.1 above could represent a compatible relationship between **P** and **Q**. But if **Q** includes a question such as "Why did *Mary* win the Award, but not *Andrew*?", then this is an incompatible relationship because there is no way *Andrew* could have won the Writer's award since he did not compete, at least going by only the information present here.

Fact	Foil
Peter's loan was denied	Why was Peter's loan application denied? (*implicit*) What must Peter do to get his loan approved? (*Explicit*)

Table 3.2: *Fact & Foil: Credit Loan*

Using the credit approval scenario, Table 3.2 shows a fact and foil table displaying a sample of possible contrastive explanations that Peter would require if his loan application was denied.
In general, contrastive explanations are sought as a response to specific counterfactual scenarios, where the focus shifts from inquiring why event **P** occurred to understanding why event **P** took place instead of an alternative event **Q**. Consequently, with contrastive explanations, our interest lies in offering an answer to the question, "Why does π occur instead of δ?", where π represents the prediction for a given instance i, and δ represents an alternative (desired) prediction.

3.2.1 Limitations of counterfactual explanations

Like all concepts in Artificial intelligence, counterfactual explanations have limitations that are seldom explicitly acknowledged or articulated. Therefore, It is the duty of the AI administrator to be aware of possible limitations in order to guard against bias and misuse. In the paper *Counterfactual Explanations of Machine Learning Predictions: Opportunities and Challenges for AI Safety* [50], *S. Kacper & F. Peter* discussed a few of these, such as:

- They can be actionable, but they are not causal,
- They are local explanations whose insights must not be generalised.
- They may need a context for viability,
- Selecting the right one amongst multiple similar counterfactuals may be challenging,
- If misinterpreted, adverse effects may occur to an AI system, causing indirect harm to involved individuals.

On critical examination, the author concurs that counterfactuals should be alternatives to a fact based on a cause, not causal. They are by nature, local explanations per observational unit and must not be generalised to all observations in a data set. Of course, misinterpretation of counterfactual explanations built into AI systems would likely cause harm or at a minimum cause bias and unfairness.

3.3 Contrastive learning

Having explored causal and contrastive explanations, let us now consider a concept for modelling contrastive explanations. In the paper, *Contrastive Representation Learning: A Framework and Review* [51], *Le-Khac et al (2020)* describe contrastive or representational learning as:

> "*Representation learning refers to the process of learning a parametric mapping from the raw input data domain to a feature vector or tensor, in the hope of capturing and extracting more abstract and useful concepts that can improve performance on a range of downstream tasks.* [51]"

That definition may seem a bit abstract or too technical depending on the reader's experience, but notwithstanding, the author shall in a few paragraphs, attempt to decipher the essence of Contrastive Learning. Before going into any technicalities, a little bit of human comprehension may be useful. Recall that as humans, we often do not zoom into details, rather we recognize general features or attributes much better. A case in point is the 5-pound note. I bet that if we were asked to draw a 5-pound bill, most of us will draw a rather vague, flat image, lacking the unique design and features of this bill, even though we may have many years of visual experience with it. Our inability to readily reproduce the intricate and unique features of this bill does not mean that we do not recognize the bill if we saw one. It simply means that we can readily identify the general features that distinguish this bill from the rest. This here, is the very essence of Contrastive Learning. To teach an algorithm (usually a DNN) the general distinguishing features of a class of data points, such that the model can generalize better to future unseen data, having learnt the distinct representations embedded in that class.

Le-Khac et al (2020) also mentioned 4 representational learning approaches applicable to machine learning models including contrastive learning algorithms.

1. Generative
2. Discriminative
3. Supervised
4. Unsupervised

While most representation learning methods assume that a good representation captures the underlying factors that explain variations in the data x, they differ in the learning process. However, *Generative approaches* learn representations by modelling the data distribution $p(x)$,

on the assumption that a good model $p(x)$ should generate realistic data samples which capture the underlying structure related to the target variable y. This approach often uses Baye's Rule for evaluating the conditional probability $p(y|x)$ for some generative tasks on variable y.
On the other hand, *Discriminative approaches* learn a representation by directly modelling the conditional probability $p(y|x)$ with a parameterised model that takes as input the data sample x and outputs the target y [51].
Without a doubt, until recently, the most successful application of deep learning has been supervised learning. Where properly annotated data is used to train Neural Networks. A few prominent use cases in recent times include CIFAR-10 [52] and ImageNet [53] competitions. The predicament associated with supervised learning becomes apparent as deep learning model architectures expand, revealing the substantial hindrance imposed by the requirement for labelled data. Consequently, an alternative training paradigm known as **Self Supervised Learning** (SSL) [54] has witnessed a notable surge in popularity recently. At the helm of this training approach lie contrastive learning methods [51].
SSL is usually applied via unsupervised learning. For example, a neural network may be taught different views of the same image and evaluated on its ability to learn these distinct views contrastively to other views. Afterwards, the same Neural Network may be fine-tuned for a specific image classification task. This approach has often outperformed traditional methods. A worthy case in point is the *SIMCLR* image classification framework [55], developed by Google Engineers, who combined SSL, unsupervised machine learning and contrastive learning methods.

Irrespective of the aim or approach, all contrastive learning algorithms have a largely similar framework as seen in Fig 3.2. This framework builds on top of the work of *Chen et al, Feb. 2020, pp. 1–20* as cited in [51]. The underlying concept is intuitive as seen below, aptly summarized by the author...

1. For every sample query $p(q)$ drawn from a dataset Q,

2. A similarity distribution $p(K^+|q)$ and a dissimilarity distribution $p(K^-|q)$ is drawn, based on a similarity function $A(.)$. Thus, data points close to q belong to the similarity distribution class K^+, while the other data points belong to the dissimilarity class K^-

3. Next, is the encoding or vectorization stage often performed by a feature encoder $e(.)$, whose job is to extract the feature vectors v from all augmented data distributions.

4. v is then fed as input to the *Head*, a Neural Network $h(.)$, whose function is to obtain a metric embedding $z = h(v)$. This has the final output of distilling both distributions into respective unit vectors.

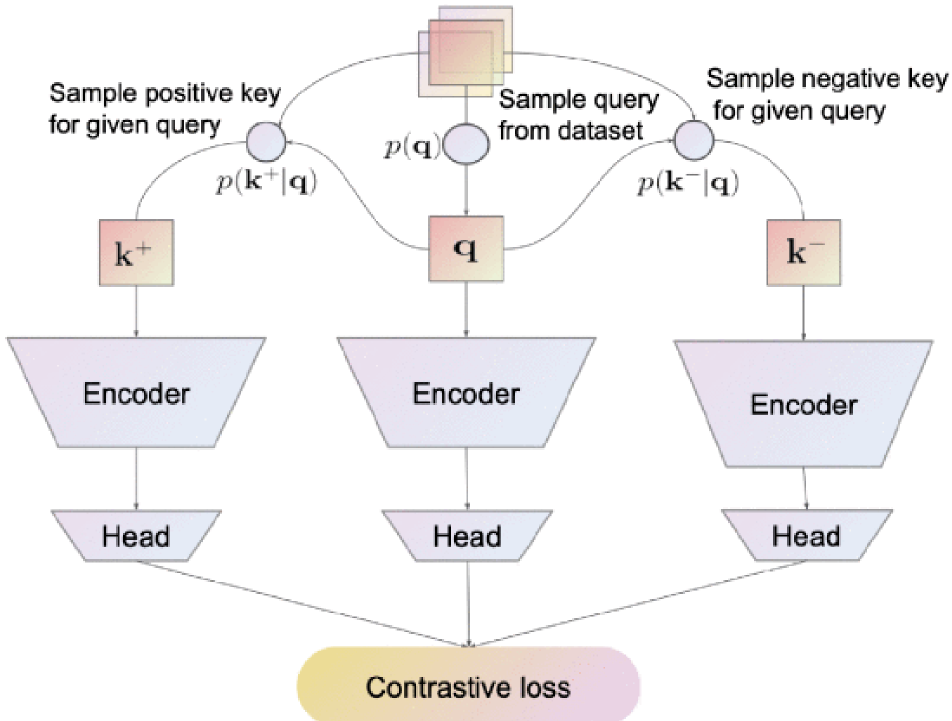


Figure 3.2: *The Contrastive Representation Learning Framework* [30]

5. Finally, batches of the metric embedding pairs are fed to a contrastive loss function, which encourages distances from similar distributions to be small and distances from different distributions to be large.

"***The goal of contrastive learning is very simple: The representation of 'similar' samples should be mapped close together, while that of 'dissimilar' samples should be further away in the embedding space [51].***"

3.3.1 Contrastive problem formulation:

- Given a training data $X\{x, x_1, x_2 ... x_n\}$ drawn from an independent and identical distribution, where each sample belongs to one of K latent classes $C_1, C_2 ... C_K$,
- Any given data point x fed to a stochastic data augmentation function $A(x)$, results in a few new views or variations, $A(x)$: $\bar{x}_i$, $\bar{x}_j$,
- We assume that $x \in A(x)$ for any x, and samples from different latent classes never transform into the same augmented sample, i.e. $A(C_K) \cap A(C_1) = \varnothing$

Contrastive Learning aims to learn an encoder $f(x)$, such that positive samples $p(K^{+}|x)$ are closely aligned. In order to make the samples from different latent classes $p(K^{-}|x)$ far away from each other.

Different loss functions exist for evaluating contrastive learning. Below is the general form of non-parametric classification loss and its variants, such as *InfoNCE* and *NT-Xent*, deployed by *Chen et al, Feb. 2020* as cited in [51].

$$\mathcal{L}_i = -log\frac{exp(\mathbf{z}_i{}^T\mathbf{z}'_i/\tau)}{\sum_{j=0}^{K} exp(\mathbf{z}_i.\mathbf{z}'_j)/\tau}$$

Figure 3.3: *Non-Parametric Classification Loss*

Where $\mathbf{z}^T\mathbf{z}'$ is the dot product between two vectors and τ is a temperature hyper-parameter that controls the sensitivity of the product. The summation in the denominator is calculated over a minibatch comprising of one positive pair and K negative pairs. Thus, To minimize the loss, the dot product in the numerator, which measures the similarity between representations from the same pair, is maximized, while simultaneously minimizing the similarity among all negative pairs in the denominator [51].

3.4 Contrastive Explanation Method (CEM)

3.4.1 CEM Structure

This section elaborates on the proposed structure of CEM. Ideally, CEM should be applied to the prediction of a Neural Network model. It aims to justify the forecast and to better inform the user about the features and minimal values that are most critical to the model's prediction. While the focus of CEM is on the output of a model given a set of examples, the author wishes to highlight that CEM is only valuable if adequate data pre-processing techniques have been applied to reduce untidiness and possible bias. These steps may include data wrangling, cleaning, feature encoding, dealing with invalid values, missing values imputation, and feature engineering. Otherwise, the model's predictions may remain flawed.
CEM was developed by IBM researchers in 2018 [9], and the Python tool kit was released in October 2020. It is part of the robust models for explainable machine learning, within the **IBM AIX360** explainable machine learning toolkit [38].

CEM is built upon the following structure:

- Consider the feasible data space $\mathcal{X}$, and let $(\mathbf{x_0}, \mathbf{t_0})$ represent an example where $\mathbf{x_0}$ belongs to $\mathcal{X}$ and $\mathbf{t_0}$ is the inferred class label obtained from a neural network model.
- The modified example $\mathbf{x}$, derived from $\mathbf{x_0}$, is defined as: $\mathbf{x} = \mathbf{x_0} + \delta$, where δ represents a perturbation applied to $\mathbf{x_0}$. Thus, $\mathbf{x} \in \mathcal{X}$.
- $PP = f(\delta|\mathbf{x_0}^+)$, and $PN = f(\delta|\mathbf{x_0}^-)$, where $f(.)$ is an optimisation function over δ given $\mathbf{x_0}$, and PP/PN are pertinent $\pm$ values.
- To ensure that $\mathbf{x} \approx \mathbf{x_0}$, we evaluate the closeness of $\mathbf{x}$ to $\mathbf{x_0}$ using an Autoencoder function $AE(.)$, $AE(\mathbf{x})$.

3.4.2 CEM Overview

In sub-section 3.1.1 above, we observed the causal relationships of contrastive explanations, within the mathematical framework, via a DAG graph. In this section, we shall observe how CEM can be applied to the prognostications of a Neural network, given a set of examples X, often a labelled data set. Perhaps, It is imperative to draw the reader's attention to the concept of ***Feature Attribution***. In the paper *AI explainability white paper* [37], *Google Cloud Engineers, (2019)* have it that "*Feature attributions provide a signed per-feature attribution score proportional to the feature's contribution to the model's prediction.*"
In essence, this technique enables the comprehension of the factors that influenced a specific prediction made by the model. To illustrate, consider a loan approval model. The feature attribution score can provide insights into how a person's account balance and credit score played a significant role in determining the model's prediction for their loan application.

Perhaps, the most robust Feature Attribution technique is SHAP [56]. SHAP values *Completeness*, *Symmetry* and *Additivity*. offer an interpretive framework to assess the influence of a specific feature value in contrast to the prediction that would be made if the feature assumed a reference or baseline value.

By contrast, Feature Attribution scores can tell us what features are most informative to a given prediction, while The Contrastive Explanation Method goes a step further to identify Pertinent Negatives (PN) and Pertinent Positives (PP) to justify the classification of an input by a Black-box classifier such as a DNN, thereby providing further information that advances the local Post-hoc explanation for that observation [9]. This means CEM can tell us what values of certain features are minimally but critically absent to justify the model's prediction in fidelity to a given class K^+, as well as the minimum values beneath which, the prediction would likely have been of a different class K^-.

D. Amit et al (2018) in the paper *Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives*, laid bare the concept and focus of the Contrastive Explanation Method. The authors of the paper deduced that CEM is a novel technique that has not been explicitly addressed by the existing explanation methods employed for interpreting predictions generated by Neural Networks. CEM attempts to resolve two highly fundamental factors that could justify and advance Local Post-Hoc explanations from the predictions of Black-box models.

- ***What is minimally but critically absent?***
- ***What is minimally sufficient?***

These two factors are known as Pertinent Negatives (PN) and Pertinent Positives (PP). Together, they constitute the critical pillars of CEM.

"*Lawrence is the **Dark** guy with **Side-beards**, who doesn't wear **Ear-studs**...*"

Explanations such as the exemplary quote above, are commonly employed by people to discern individuals or objects of interest. In this instance, attributes like ***Dark*** and ***Side-beards*** partially describe *Lawrence*, yet the absence of ***Ear-studs*** plays a crucial role in providing a complete identification and distinguishing him from another individual, such as *Mike*, who shares similar characteristics as *Lawrence*, but wears Ear-studs. Humans frequently employ such contrasting details when aiming to provide precise explanations. This approach makes practical and rational sense as commonly found in human communications. Thus, it appears that the purpose of CEM is to improve the explanations of Black-box models such that they are as humanly relevant as possible. This is done by adopting a minimalist approach that focuses on those traits that are characteristically prime to the input, given similar inputs. In this scenario, the identified contrastive characteristics (***Dark*** and ***Side-beards***) are by no means an exhaustive list of characteristics that *Lawrence* possesses, but they are sufficient to distinguish him from the closest person(s) that may look like him, but wears ***Ear-studs***.

Figure 3.4 above depicts the classification details done by CEM, Layerwise Relevance Propagation (LRP), and Locally Interpretable Model-agnostic Explanations (LIME) on the

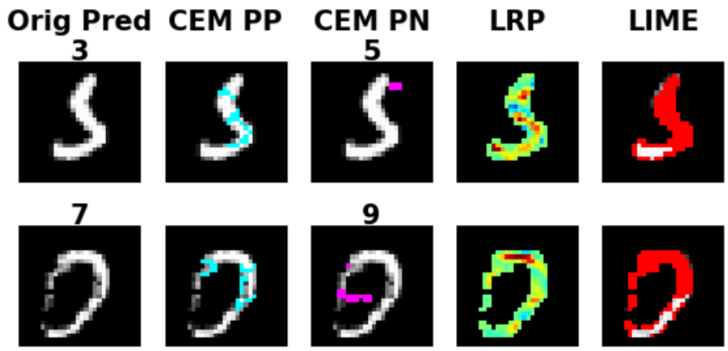


Figure 3.4: *CEM versus LRP and LIME on MNIST* [9]

popular MNIST dataset [9]. For the sake of this dissertation, the writer shall only comment on the CEM values.

What is most striking about this image is the depiction of PN and PP by CEM, to justify its prediction. As visible in the top row of the *CEM PP* column, we can see the faint cyan curve that traces the central curve of the digit **3**, which must be *minimally present* and sufficient to indicate that the digit is a **3** or a **5**, based on some probability confidence score and not likely a **1**, for instance. This is the Pertinent Positive for justifying the rightful classification of this digit as a **3** or a **5**. In the same vein, the top row of the *CEM PN* column remarkable depicts the part of the digit that is *minimally but critically absent*, such that the model is now more confident that due to this absence as denoted by the small horizontal pink tick, the digit is most likely a **3** and not a **5**, rightfully classifying the image as a **3**.

Therefore, the explanation for the digit in row 1 above would be:

"*The row 1 digit is a 3 because the cyan pixels (shown in column 2) are present and the pink pixels (shown in column 3) are absent*".

According to *D. Amit et al (2018)*, this second part of the explanation is critical for accurate classification. Yet, it is not highlighted by any of the other state-of-the-art interpretability methods such as LRP or LIME, hence underscoring the uniqueness of the CEM method.

Largely, what can be inferred from CEM is the fact that these contrastive facts should not be viewed as an exhaustive compilation of all conceivable characteristics that must be lacking in an input, to differentiate it from all other classes to which it does not belong. Instead, they represent a minimal set of features that assist in distinguishing it from the nearest class to which it does not belong.

3.4.3 Pertinent Negatives (PN)

"*What is minimally but critically absent?*"

$$\min_{\delta \in \mathcal{X}/\mathbf{x_0}} c \cdot \mathrm{f_{neg}}\kappa(\mathbf{x_0}, \delta) + \beta\|\delta\|_1 + \|\delta\|_2^2 + \gamma\|\mathbf{x_0} + \delta - \mathrm{AE}(\mathbf{x_0} + \delta)\|_2^2$$

Figure 3.5: *PN Optimisation Function*
[9]

Given the feasible data space $\mathcal{X}$, let $(\mathbf{x_0}, \mathbf{t_0})$ represent an example where $\mathbf{x_0}$ belongs to $\mathcal{X}$ and $\mathbf{t_0}$ is the inferred class label obtained from a neural network model. In the context of pertinent negative analysis, the focus lies on identifying the elements that are absent in the model's prediction. The set of missing components with respect to a given natural example $\mathbf{x_0}$, is denoted as $\frac{\mathcal{X}}{\mathbf{x_0}}$. Thus, the objective is to discover an interpretable perturbation $\frac{\delta \in \mathcal{X}}{\mathbf{x_0}}$, which allows for an examination of the disparity between the predictions of the most probable classes:

$$(argmax_i[Pred(\mathbf{x_0})]i) \;\; \& \;\; (argmax_i[Pred(\mathbf{x_0} + \delta)]i)$$

Therefore given $(\mathbf{x_0}, \mathbf{t_0})$, CEM finds the pertinent negative by solving an optimisation problem consisting of the following components:

- Loss function (L):
  $f_{\text{neg}}\kappa(\mathbf{x_0}, \delta) = \max\left\{[Pred(\mathbf{x_0} + \delta)]_{\mathbf{t_0}} - \max_{i \neq \mathbf{t_0}}[Pred(\mathbf{x_0} + \delta)]_i, -\kappa\right\}$,

  where $[Pred(\mathbf{x_0} + \delta)]_i$ is the i-th class prediction score of $\mathbf{x_0} + \delta$.
- Elastic Net Regularizer (ENR):
  $\beta\|\delta\|_1 + \|\delta\|_2^2$
- Autoencoder $L2$ Error Evaluation:
  $\|\mathbf{x_0} + \delta - AE(\mathbf{x_0} + \delta)\|_2^2$

The utilized hinge-like L function is designed to encourage the modified example $\mathbf{x}$ to exhibit a top-1 predicted class difference from the original example $\mathbf{x_0}$. While ENR is used for efficient feature selection in high-dimensional feature spaces. Finally, an Autoencoder $AE(.)$ is applied to evaluate the $L2$ reconstruction error of $\mathbf{x}$. This is relevant to ensure that $\mathbf{x} \approx \mathbf{x_0}$.
Therefore, Fig 3.5 shows the PN optimisation function by combining the 3 components above.

3.4.4 Pertinent Positives (PP)

"*What is minimally sufficient??*"

Given the feasible data space $\mathcal{X}$, let $(\mathbf{x_0}, \mathbf{t_0})$ represent an example where $\mathbf{x_0}$ belongs to $\mathcal{X}$ and $\mathbf{t_0}$ is the inferred class label obtained from a neural network model. In the context of pertinent positive analysis, the focus is on finding the critical features that are readily present in the input $\mathbf{x_0}$. Therefore, considering a natural example such as $\mathbf{x_0}$, we can represent the set of its constituent elements by $\mathcal{X} \cap \mathbf{x_0}$.
The focus here lies on finding an interpretable perturbation δ given that $\delta \in \mathcal{X} \cap \mathbf{x_0}$, such that after removing it from $\mathbf{x_0}$ we would obtain the following:

$$argmax_i[Pred(\mathbf{x_0})]i \quad = \quad argmax_i[Pred(\delta)]i$$

In other words, both $\mathbf{x_0}$ and δ will yield the same top-1 predicted class $\mathbf{t_0}$, suggesting that the perturbation δ, which has been removed, accurately reflects the model's prediction for $\mathbf{x_0}$. The optimisation problem for PP is identical in structure to that of PN above. Except for the fact that the loss function for PP: $L = f^{\kappa}_{\text{pos}}$, now focuses on the perturbation δ with respect to the existing class labels, as seen below...

$f^{\kappa}_{\text{pos}}(x_0, \delta) = \max\{\max_{i \neq t_0}[\text{Pred}(\delta)]_i - [\text{Pred}(\delta)]_{t_0}, -\kappa\}$

Meaning, for any given confidence $\kappa \geq 0$, the loss function f^{κ}_{pos} is minimized when $([Pred(\delta)]_{\mathbf{t_0}}) - (\max_{i \neq t_0}[\text{Pred}(\delta)]_i) \geq \kappa$

Taking into consideration the loss function and the previously mentioned components, the optimisation function for finding PP is denoted as:

$$\min_{\delta \in X \cap x_0} c \cdot f^{\kappa}_{\text{pos}}(x_0, \delta) + \beta\|\delta\|_1 + \|\delta\|_2^2 + \gamma\|\delta - \text{AE}(\delta)\|_2^2$$

Figure 3.6: *PP Optimisation Function*
[9]

In the next chapter, the author shall apply this method to a real-life data set using the IBM AIX360 [38] explainable machine learning toolkit.

Chapter 4

Methodology

In this chapter, as mentioned earlier, the author will utilize the Contrastive Explanation Method (CEM) on real-life data, which involves a substantial focus on coding. However, the author has made efforts to include code cells in the dissertation's appendix section and strives to provide comprehensible explanations for each step throughout the chapter. This chapter builds upon the insights gathered over the previous chapters and applies CEM to improve the explanations from a binary classifier.

4.1 Exploratory Data Analysis (EDA)

4.1.1 The data:

In this experiment, the author shall use the **Credit One Bank**, loan data, which is freely available on Kaggle [57]. Credit One Bank is an online-only bank that exclusively offers credit cards. It is a technology and data-driven financial services company that is based in Las Vegas, U.S.A. The bank offers American Express and Visa credit cards to millions of customers across the United States [58]. The loan data set contains variables that could be largely grouped into 3 categories:

- **Customer info**: Examples include *Age*, *Gender*, *Dependents*,
- **Job/Personal finance**: Such as *Status checking account*; *Savings account*,
- **Loan data**: Such as *Credit amount* and *Credit purpose*.

This data is real-life and has been fully anonymized to protect the privacy of customers. Yet, it holds real values used to train a classifier to predict if a customer/loan applicant is likely to default in re-paying their loan if advanced, a problem framed as a **Binary-Classification** task. The data

contains the credit card application details of 5000 customers, across 22 different features or variables that offer insights into the personal and financial circumstances of the applicants. The data and its attendant data dictionary (A document showing the respective variables in the data set and their meanings) can be downloaded via Kaggle [57].
This data was chosen for its extensive coverage of features commonly required for credit approval across the 3 listed categories. It accurately reflects real-world situations where training data often exhibits irregularities and imbalances, with regular data prevailing over anomalous instances. This poses a significant challenge in developing AI solutions to address anomalies like credit card theft or spam.

[57]

Table 4.1: *Cross-section of Credit-One Bank Loan Data Variables*

Variable Name	**Summary**	**Data Type**
PURPOSE_CREDIT_TAKEN	Options include new-car, used-car, furniture, repairs, education...	Categorical
MARITAL_STATUS_GENDER	Marital/Gender status such as Divorced/separated, Single...	Categorical
HOUSING	Accommodation status of applicant (A151 = Rent, A152 = Own, A153 = For free)	Categorical
MARRIAGE	Marital status of the individual (1 = married, 2 = single, 3 = others)	Categorical
AGE	Age of the individual in years	Numeric
DEPENDENTS	Number of dependents	Numeric
FOREIGN_WORKER	Foreign work status (A201 = Yes, A202 = No)	Categorical
CURRENT_ADDRESS_Yrs	Number of years at current address	Numeric
DEFAULT_ON_PAYMENT	Whether the individual defaulted on the payment in the past. This is the Target variable (Yes = 1, No = 0)	Categorical

Data processing was done within the Anaconda-Python environment. The data had no missing values or duplicates. The author processed the data by cleaning column names and arranging similar columns contiguously to create meaningful tables in line with the tidy data workflow proposed by W. Hadley [59]

4.1.2 Feature selection:

It is widely known that often, Data Scientists spend about 70% of the time wrangling and processing data. This process is usually iterative and encompasses several activities, chief amongst which is Feature-selection. While there may be various approaches to employ, the end goal is usually the same: To find the best features whose patterns a given machine learning algorithm may easily approximate, within the given data space $\mathcal{X}$.
Feature selection is specific to the data. Strategies that work for one may not work for another data set. For example, the Credit One Bank, loan data set has over 50% of categorical variables, much of which also have multiple categorical values embedded. This creates an immediate challenge, which can only be surmounted with aid from the Data Dictionary and concise feature selection methods.

To solve this challenge, the author one-hot-encoded these variables and created a delete function that computes the relationship between each categorical-feature/value and the target variable, such that $(t < 30) or (t > 60)$ where t is the percentage threshold of defaulting values per categorical-feature.

```
def delete_categorical_feature(categorical_cols, llim, hlim):
""" This method takes a list of categorical columns,and returns a list
of feature attributes to delete based on set threshold

@param categorical_cols: A dataframe of categorical variables
@param llim: The lower limit, default=0.3
@param hlim: The higher limit, default=0.6
"""

    delete_list = []
    for x in categorical_cols:
        default_df = loan_df.groupby(x)['default_on_payment'].value_counts(normalize=True).to_frame()[1::2]
        delete_feature = default_df[(default_df.default_on_payment >= llim) & (default_df.default_on_payment <= hlim)]
        delete = [delete_feature.index[i][0] for i in range(len(delete_feature))]
        for i in range(len(delete)):
            delete[i] = x + '_' + delete[i]
        delete_list += delete
    return delete_list

delete_list = delete_categorical_feature(cat_cols, llim, hlim)
```

Listing 4.1: Categorical features selection function

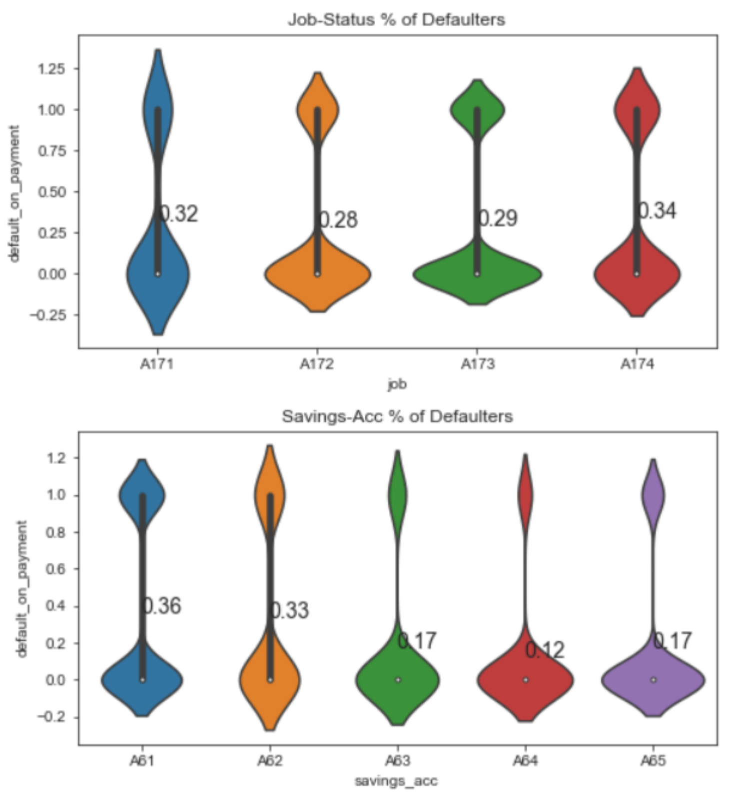


Figure 4.1: *A few categorical variables with multiple categorical values...*

4.1.3 Model training and evaluation:

The data was grossly imbalanced in favour of the non-defaulting class at a ratio of $0.7 : 0.3$. To balance the data, the following methods were tested:

- Synthetically oversampling the minority class
- Applying class weights to even the training parameters of the model

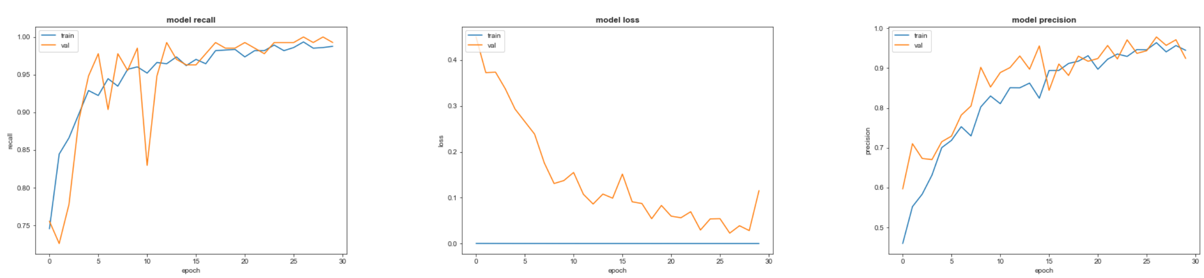
The class weights method proved the better of both with $Recall : 0.97$ and $Precision : 0.94$. A Sequential model from the *Keras* library was used, with 3 Dense layers, each having the RELU activation function, and 256 neurons, and a final dense layer with the *Sigmoid* function, suitable for binary classification tasks such as this. Between the last 3 Dense layers are 2 Dropout layers, with a 30% drop-out rate, to prevent over-fitting of the model.

```
model = keras.Sequential(
[
    keras.layers.Dense(
        256, activation="relu", input_shape=(train_features.shape[-1],)
    ),
    keras.layers.Dense(256, activation="relu"),
    keras.layers.Dropout(0.3),
    keras.layers.Dense(256, activation="relu"),
    keras.layers.Dropout(0.3),
    keras.layers.Dense(1, activation="sigmoid"),
]
)

```

Listing 4.2: Model architecture

The model was trained for 30 epochs, with a batch size of 64 training examples, and a hold-out validation set for real-time evaluation. As mentioned earlier, the model achieved $Recall : 0.97$ and $Precision : 0.94$. The author favoured these metrics over the more common accuracy score, due to the high imbalance of the data. This exercise is attached in the appendix section. Kindly see the models performance on the training and validation sets below:



(a) *Recall* **(b)** *Loss* **(c)** *Precision*

Figure 4.2: *The model training metrics: yellow=val, blue=train curves*

4.1.4 CEM Explanations:

In this section, the author shall demonstrate the computation of contrastive explanations for a credit application, from the Credit One Bank loan data set. Recall from sections 3.4.3 and 3.4.4 that CEM outputs a contrastive explanation by decomposing a model's prediction into two distinct parts Pertinent Negatives (PN) and Pertinent Positives (PP), based on some perturbation function on the prediction. For instance, consider the feasible data space $\mathcal{X}$, and let $(\mathbf{x_0}, \mathbf{t_0})$ represent an example where $\mathbf{x_0}$ belongs to $\mathcal{X}$ and $\mathbf{t_0}$ is the inferred class label obtained from a neural network model. The modified example $\mathbf{x}$, derived from $\mathbf{x_0}$, is defined as: $\mathbf{x} = \mathbf{x_0} + \delta$, where δ represents a perturbation applied to $\mathbf{x_0}$. Thus, $\mathbf{x} \in \mathcal{X}$ [9]. The programming guide for a binary classification task using CEM involves the following broad steps:

1. Process and Normalize the training data
2. Define and Train a Neural network Classifier
3. Compute Contrastive Explanations for a few applicants.

Technical specifications:

CEM is an algorithm within the IBM AIX360 toolkit. This toolkit contains about 10 impressive algorithms offering various types of explanations across the global/local/intrinsic/post-hoc landscape. CEM requires very specific versions from specific Python libraries. There is a *PyPI* option for Python <`pip install aix360`>, but it turned out that this version was not fully updated, leading to more than a few Runtime errors. Therefore, the author chose the Anaconda install version, which is actually the recommended style as stated by the CEM repo coordinators in Github. Its setup is quite technical and the author found it somewhat stressful to install.

Figure 4.3: *CEM installation steps using Anaconda...*

Create Venv	MK Dir	CD Dir	Clone AIX360	Run setup.py

From fig 4.3, setting up CEM using Anaconda involves the following:

- Run the Anaconda terminal and create a Virtual Environment specific for Python 3.6, then activate it:

```
conda create --name aix360 python=3.6
conda activate aix360
```

- Make a new directory and Change to it:

```
mkdir aix360
cd aix360
```

- Clone the AIX360 Trusted Github repo: `<git clone https://github.com/Trusted-AI/AIX360>`
- Finally, navigate to the CEM dir and run the setup file:

```
cd AIX360
pip install -e .[contrastive]
```

The installation guide can be found at the IBM AIX360 Trusted Github repo [60].

4.1.5 Pertinent Negatives (PN) Algorithm

PNs identify minimal features altering the classification. E.g., increasing years at current address may change reject to accept. Section 3.4.3 demonstrates using elastic norm to minimize these changes, as well as an Autoencoder [60]. See below, the PN algorithm in Python

:

```
idx = def_idx[0]

X = x_test[idx].reshape((1,) + x_test[idx].shape)
print("Computing PN for Sample:", idx)
print("Prediction made by the model:", model.predict_proba(X))
print("Prediction probabilities:", class_names[np.argmax(model.
predict_proba(X))])
print("")

mymodel = KerasClassifier(model)
explainer = CEMExplainer(mymodel)

arg_mode = 'PN' # Find PN
arg_max_iter = 1000 # Max iter for optimal PN given params
arg_init_const = 10.0 # Initial coef for main loss term that encourages
class change
arg_b = 9 # No. of updates to coef
arg_kappa = 0.2 # Minimum confidence gap between PN & PP class
probabilities
arg_beta = 1e-1 # Controls sparsity of the solution (L1 loss)
arg_gamma = 100 # Controls adherence to auto-encoder
my_AE_model = None # Pointer to an auto-encoder
arg_alpha = 0.01 # Penalizes L2 norm of the solution
```

```
    arg_threshold = 1 # Auto off features <= arg_threshold if arg_threshold <
    1
    arg_offset = 0.5 # the model assumes data normalized in [-0.5, 0.5] range

    # Find PN for the applicant
    (adv_pn, delta_pn, info_pn) = explainer.explain_instance(X, arg_mode,
                                                              my_AE_model,
    arg_kappa, arg_b,
                                                              arg_max_iter,
    arg_init_const, arg_beta, arg_gamma,
                                                              arg_alpha,
    arg_threshold, arg_offset)

```

Listing 4.3: PN algorithm

4.1.6 Pertinent Positives (PP) Algorithm

PPs identify minimal features sufficient for a specific classification. E.g., a 25K credit, not 80K, may be sufficient. As illustrated in Section 3.4.4, an elastic norm is used to reduce the information required.

The below codes which can be simply wrapped into a function, compute PP for an applicant [60].

```
idx = ndef_idx[0]

X = x_test[idx].reshape((1,) + x_test[idx].shape)
print("Computing PP for Sample:", idx)
print("Prediction made by the model:", class_names[np.argmax(model.
    predict_proba(X))])
print("Prediction probabilities:", model.predict_proba(X))
print("")


mymodel = KerasClassifier(model)
explainer = CEMExplainer(mymodel)

arg_mode = 'PP' # Find PP
arg_max_iter = 1000 # Max iter for optimal PP given params
arg_init_const = 10.0 # Initial coef for main loss term that encourages class
    change
arg_b = 9 # No. of updates to coef
arg_kappa = 0.2 # Minimum confidence gap between PN & PP
```

```
arg_beta = 10.0 # Controls sparsity of the solution (L1 loss)
arg_gamma = 100 # Controls adherence to auto-encoder
my_AE_model = None # Pointer to an auto-encoder
arg_alpha = 0.1 # Penalizes L2 norm of the solution
arg_threshold = 0.0 # Auto off features <= arg_threshold if arg_threshold < 1
arg_offset = 0.5 # the model assumes data normalized in [-0.5, 0.5] range
(adv_pp, delta_pp, info_pp) = explainer.explain_instance(X, arg_mode,
                                                         my_AE_model,
                                                         arg_kappa,
                                                         arg_b,
                                                         arg_max_iter,
                                                         arg_init_const,
                                                         arg_beta,
                                                         arg_gamma,
                                                         arg_alpha,
                                                         arg_threshold,
                                                         arg_offset)

```

Listing 4.4: PP algorithm

4.1.7 Findings & Insights

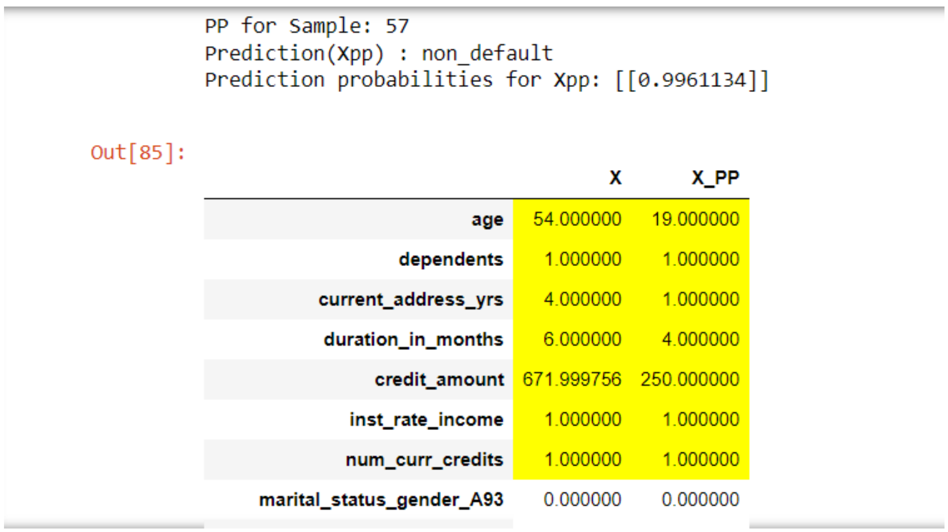
The first thing to note is the identical design of the algorithms for both PN and PP. In fact, they are identical, aside the values of certain parameters such as *arg_beta*, *arg_thresholds*. The only difference in implementation is the computation of the loss function for PP, which focuses on the perturbation δ with respect to the class labels: Section 3.4.4

The author shall now compute the PP insights for a given applicant from the Credit One Bank, loan data. This applicant shall be referred to as *Person 57*. The applicant's prediction from the binary classifier is fed as input the the PN algorithm above. *Person 57* is one of the applicants classified as a non-defaulter. Meaning, they belong to the positive class that should be advanced credits. Kindly see Fig 4.4a below.

The insights from *Person 57*'s credit application indicate that, they would have been accepted with only three variables unchanged: *Dependents*, *Installment-rate-Income*, and *number of current credits*. There was also a notable reduction in Credit Amount, and a slight decrease in the Duration in months from 6 to 4. This means that even if the credit amount applied for was reduced to 250 from 671, their credit application would still be approved. It is important to state here that explanations may vary due to equivalent values in local minima.

4.1.8 Interactive Visualization:

As mentioned earlier, the author used the Google What-If-Tool [12] for interactive visualization of the model. This can be demonstrated during the Viva. Do note that there are limitations in interactivity pertaining to contrastive explanations. This is because contrastive explanations are still being devoloped and they are quite subjective, depending on the user's requirements. most visualization tools focus more on the model parameters and how these can alter predictions.

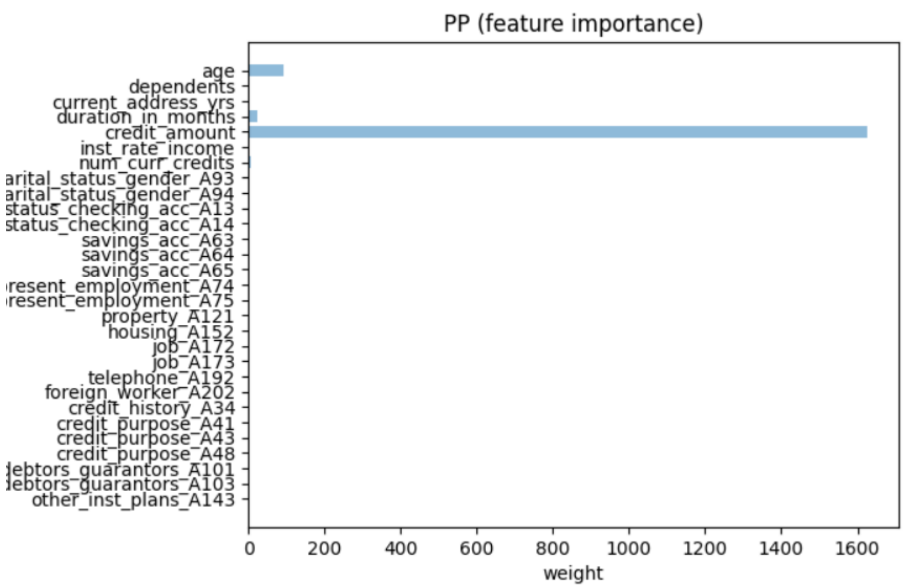


```
PP for Sample: 57
Prediction(Xpp) : non_default
Prediction probabilities for Xpp: [[0.9961134]]
```

Out[85]:

	X	X_PP
age	54.000000	19.000000
dependents	1.000000	1.000000
current_address_yrs	4.000000	1.000000
duration_in_months	6.000000	4.000000
credit_amount	671.999756	250.000000
inst_rate_income	1.000000	1.000000
num_curr_credits	1.000000	1.000000
marital_status_gender_A93	0.000000	0.000000

(a) *PP Scores*



(b) *Feature Importance*

Figure 4.4: *Feature Importance & PP Scores: Person 57*

Chapter 5

Conclusion

5.1 Discussion

In this dissertation titled ***Improving Credit Approval Decisions with Explainable Machine Learning***, the author discussed the current state of Artificial Intelligence (AI), and Explainable AI (XAI), given their applications to credit lending, and how XAI can improve credit approval explanations made to customers via the Contrastive Explanation Method (CEM).

The initial chapters established the thesis groundwork, exploring AI and XAI before focusing on local post-hoc explanations for customers through contrastive learning. The author also examined the challenges of XAI, particularly the human-in-the-loop effect, and drew connections between the necessity of enhancing Black-box explanations and ensuring thorough algorithm testing.

In Chapter Three, the author extensively examines key strategic concepts, namely causal explanations and contrastive explanations, as the foundation for the contrastive explanation method. The author implements causal modeling using a Directed Acyclic Graph (DAG) and provides a comprehensive exploration of the contrastive learning framework. The chapter culminates in the formalization of the contrastive problem formula.

Chapter Four presents the author's practical implementation of the Contrastive Explanation Method (CEM) to enhance credit approval decisions using a binary classification model. The author covers the entire data processing life cycle, including data wrangling, feature selection, model training, and evaluation. Through the use of anonymous data, the author effectively demonstrates the computation of Pertinent Negatives and Pertinent Positives, offering improved explanations for credit applicants. This empirical analysis showcases the practical utility and effectiveness of CEM in improving credit approval decisions.

5.2 Limitation

It is important to acknowledge a few limitations in this study. While the contrastive explanation method offers valuable insights, its applicability may be constrained by certain factors, such as:

- Data availability: The abundance of unstructured data presents a challenge in structuring data appropriately for XAI. This challenge is further amplified by the diverse user needs and the different formats and types of explanations that are required.
- Technical specifications: As discussed in Chapter two, the tools and nomenclature in XAI are quite distinct. Each tool necessitates a unique technology stack or set of libraries for its execution. Consequently, this poses an immediate challenge for XAI, as the integration and utilization of diverse tools and libraries become necessary.
- Model complexity: Model complexity can impede XAI by making it more challenging to interpret predictions from DNNs that incorporate numerous variables and complex interactions. Surrogacy often fails to explain DNNs, leading to the decomposition of explanations into local, observational-specific units as observed in practice.
- Interpretability challenges: Chapter Two addressed the human reasoning challenge, recognizing that humans often think and act irrationally. This significantly impacts the handling of explanations derived from complex models, as we must consider human cognitive limitations and biases.

These limitations should be considered when interpreting the findings and applying the method in practical scenarios.

5.3 Future Work

Explainable Artificial Intelligence is still at the research stage and many solutions are not yet practicable at scale. CEM is an impressive technique that can provide credence and improve confidence in model explanations. Therefore, the author has identified three key areas that future research should focus on:

- Exploring synergies with other explainability methods and techniques, such as rule-based explanations or model-agnostic approaches, to complement CEM and provide a comprehensive understanding of model behavior.

- Addressing scalability: Developing scalable algorithms and methodologies to handle large-scale datasets and complex models, ensuring efficient and timely generation of contrastive explanations.

- Integrating domain knowledge: Incorporating domain-specific knowledge and expert insights into the CEM framework to enhance the quality and relevance of the generated explanations.

Bibliography

[1] V. Arya(2), R. K. E. Bellamy, P.-Y. Chen, A. Dhurandhar, Michael Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilovic, S. Mourad, P. Pedemonte, R. Raghavendra, J. Richards, P. Sattigeri, K. Shanmugam, M. Singh, K. R. Varshney, D. Wei, and Y. Zhang, "AI Explainability 360: Impact and Design," *arXiv:2109.12151 [cs]*, Sep. 2021, arXiv: 2109.12151. [Online]. Available: http://arxiv.org/abs/2109.12151

[2] Austin Eovito, "IBM Masterclass: AI Explainability." [Online]. Available: https://ibm-explainability.bemyapp.com/#signup

[3] S. Verma, A. Lahiri, J. P. Dickerson, and S.-I. Lee, "Pitfalls of Explainable ML: An Industry Perspective," Jun. 2021. [Online]. Available: https://arxiv.org/abs/2106.07758v1

[4] European Commission, "High-level expert group on artificial intelligence | Shaping Europe's digital future," 2022. [Online]. Available: https://digital-strategy.ec.europa.eu/en/policies/expert-group-ai

[5] U. F. T. Commission, "Aiming for truth, fairness, and equity in your company's use of AI," Apr. 2021. [Online]. Available: http://www.ftc.gov/business-guidance/blog/2021/04/aiming-truth-fairness-equity-your-companys-use-ai

[6] UK Dept for Sci, Innovation, Tech and UK Office for AI, "AI regulation: a pro-innovation approach." [Online]. Available: https://www.gov.uk/government/publications/ai-regulation-a-pro-innovation-approach

[7] OECD, "OECD Legal Instruments," 2022. [Online]. Available: https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449

[8] US Deparment of Commerce, "Department of Commerce Establishes National Artificial Intelligence Advisory Committee," 2022. [Online]. Available: https://www.commerce.gov/news/press-releases/2021/09/department-commerce-establishes-national-artificial-intelligence

[9] A. Dhurandhar, P.-Y. Chen, R. Luss, C.-C. Tu, P. Ting, K. Shanmugam, and P. Das, "Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives," arXiv, Tech. Rep. arXiv:1802.07623, Oct. 2018, arXiv:1802.07623 [cs] type: article. [Online]. Available: http://arxiv.org/abs/1802.07623

[10] M. Sundararajan and A. Najmi, "The many Shapley values for model explanation," Feb. 2020, arXiv:1908.08474 [cs, econ]. [Online]. Available: http://arxiv.org/abs/1908.08474

[11] Y. Zhang, K. Song, Y. Sun, S. Tan, and M. Udell, ""Why Should You Trust My Explanation?" Understanding Uncertainty in LIME Explanations," Jun. 2019, arXiv:1904.12991 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1904.12991

[12] Google Cloud, "What-If Tool," 2022. [Online]. Available: https://pair-code.github.io/what-if-tool/

[13] Arelis Diaz, Carlos Rangel, La June Montgomery Tabron, and Joel Wittenberg, "Racial equity in financial services | McKinsey." [Online]. Available: https://www.mckinsey.com/industries/financial-services/our-insights/racial-equity-in-financial-services

[14] K. Hale, "A.I. Bias Caused 80% Of Black Mortgage Applicants To Be Denied," 2021, section: Personal Finance. [Online]. Available: https://www.forbes.com/sites/korihale/2021/09/02/ai-bias-caused-80-of-black-mortgage-applicants-to-be-denied/

[15] S. Russell and Norvig, Peter, "Artificial Intelligence: A Modern Approach, 4th US ed." Aug. 2022. [Online]. Available: http://aima.cs.berkeley.edu/index.html

[16] C.-H. Chiang and H.-y. Lee, "Can Large Language Models Be an Alternative to Human Evaluations?" May 2023, arXiv:2305.01937 [cs]. [Online]. Available: http://arxiv.org/abs/2305.01937

[17] J. S. Gans, "AI Adoption in a Competitive Market," Apr. 2022. [Online]. Available: https://www.nber.org/papers/w29996

[18] R. Chauhan, K. K. Ghanshala, and R. Joshi, "Convolutional Neural Network (CNN) for Image Detection and Recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Dec. 2018, pp. 278–282.

[19] S. Grossberg, "Recurrent Neural Networks," *Scholarpedia*, vol. 8, no. 2, p. 1888, Feb. 2013. [Online]. Available: http://scholarpedia.org/article/Recurrent_Neural_Networks,http://www.scholarpedia.org/article/Recurrent_Neural_Networks

[20] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-Art Natural Language Processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[21] K. Nair and R. Gupta, "Application of AI technology in modern digital marketing environment," *World Journal of Entrepreneurship, Management and Sustainable Development*, vol. 17, no. 3, pp. 318–328, Jan. 2021, publisher: Emerald Publishing Limited. [Online]. Available: https://doi.org/10.1108/WJEMSD-08-2020-0099

[22] F. Kong, "Application of Artificial Intelligence in Modern Art Teaching," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 15, no. 13, pp. 238–251, Jul. 2020, publisher: International Journal of Emerging Technology in Learning. [Online]. Available: https://www.learntechlib.org/p/217610/

[23] M. Y. Shaheen, "Applications of Artificial Intelligence (AI) in healthcare: A review," *ScienceOpen Preprints*, Sep. 2021, publisher: ScienceOpen. [Online]. Available: https://www.scienceopen.com/hosted-document?doi=10.14293/S2199-1006.1.SOR-.PPVRY8K.v1

[24] Bindushree, "Artificial intelligence: In modern dentistry," 2020. [Online]. Available: https://www.jdrr.org/article.asp?issn=2348-2915;year=2020;volume=7;issue=1;spage=27;epage=31;aulast=Bindushree

[25] N. Wang, Y. Liu, Z. Liu, and X. Huang, "Application of Artificial Intelligence and Big Data in Modern Financial Management," in *2020 International Conference on Artificial Intelligence and Education (ICAIE)*, Jun. 2020, pp. 85–87.

[26] X. Wu, "Application of Artificial Intelligence in Modern Vocational Education Technology," *Journal of Physics: Conference Series*, vol. 1881, no. 3, p. 032074, Apr. 2021, publisher: IOP Publishing. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1881/3/032074

[27] IBM research, "What is generative AI, what are foundation models, and why do they matter?" Mar. 2023, section: Business transformation. [Online]. Available: https://www.ibm.com/blog/what-is-generative-ai-what-are-foundation-models-and-why-do-they-matter/

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” Dec. 2017, arXiv:1706.03762 [cs]. [Online]. Available: http://arxiv.org/abs/1706.03762

[29] OpenAI, “OpenAI,” 2023. [Online]. Available: https://openai.com/

[30] Cem Dilmegani, “Large Language Model Training in 2023,” 2023. [Online]. Available: https://research.aimultiple.com/large-language-model-training/

[31] S. Kumar, L. Akoglu, N. Chawla, J. A. Rodriguez-Serrano, T. Faruquie, and S. Nagrecha, “Machine Learning in Finance,” in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, ser. KDD ’21. New York, NY, USA: Association for Computing Machinery, Aug. 2021, pp. 4139–4140. [Online]. Available: http://doi.org/10.1145/3447548.3469456

[32] R. Muñoz-Cancino, C. Bravo, S. A. Ríos, and M. Graña, “On the dynamics of credit history and social interaction features, and their impact on creditworthiness assessment performance,” *arXiv:2204.06122 [cs]*, Apr. 2022, arXiv: 2204.06122. [Online]. Available: http://arxiv.org/abs/2204.06122

[33] Mckinsey, “Designing next-generation credit-decisioning models | McKinsey,” Dec. 2021. [Online]. Available: https://www.mckinsey.com/business-functions/risk-and-resilience/our-insights/designing-next-generation-credit-decisioning-models

[34] USAGov, “Credit Reports and Scores | USAGov,” 2022. [Online]. Available: https://www.usa.gov/credit-reports

[35] CFPB_, “What is a credit report?” 2022. [Online]. Available: https://www.consumerfinance.gov/ask-cfpb/what-is-a-credit-report-en-309/

[36] N. Bussmann, P. Giudici, D. Marinelli, and J. Papenbrock, “Explainable Machine Learning in Credit Risk Management,” *Computational Economics*, vol. 57, no. 1, pp. 203–216, Jan. 2021. [Online]. Available: https://doi.org/10.1007/s10614-020-10042-0

[37] Google Cloud, “ai_explainability_whitepaper_google.pdf,” 2019. [Online]. Available: https://cerre.eu/wp-content/uploads/2020/07/ai_explainability_whitepaper_google.pdf

[38] V. Arya, R. K. E. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilović, S. Mourad, P. Pedemonte, R. Raghavendra, J. Richards, P. Sattigeri, K. Shanmugam, M. Singh, K. R. Varshney, D. Wei, and Y. Zhang, “AI

Explainability 360 Toolkit," in *8th ACM IKDD CODS and 26th COMAD*, ser. CODS COMAD 2021. New York, NY, USA: Association for Computing Machinery, Jan. 2021, pp. 376–379. [Online]. Available: http://doi.org/10.1145/3430984.3430987

[39] European Commission, "Ethics guidelines for trustworthy AI | Shaping Europe's digital future," Apr. 2019. [Online]. Available: https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai

[40] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization," 2017, pp. 618–626. [Online]. Available: https://openaccess.thecvf.com/content_iccv_2017/html/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.html

[41] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, "Local Rule-Based Explanations of Black Box Decision Systems," May 2018, arXiv:1805.10820 [cs]. [Online]. Available: http://arxiv.org/abs/1805.10820

[42] S. Hooker, D. Erhan, P.-j. Kindermans, and B. Kim, "Evaluating Feature Importance Estimates," *arXiv*, 2018. [Online]. Available: https://arxiv.org/pdf/1806.10758.pdf

[43] C. Kozyrkov, "Explainable AI won't deliver. Here's why." Oct. 2021. [Online]. Available: https://medium.com/hackernoon/explainable-ai-wont-deliver-here-s-why-6738f54216be

[44] M. Du, N. Liu, and X. Hu, "Techniques for interpretable machine learning," *Communications of the ACM*, vol. 63, no. 1, pp. 68–77, Dec. 2019. [Online]. Available: https://dl.acm.org/doi/10.1145/3359786

[45] C. Nóbrega and L. Marinho, "Towards explaining recommendations through local surrogate models," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. Limassol Cyprus: ACM, Apr. 2019, pp. 1671–1678. [Online]. Available: https://dl.acm.org/doi/10.1145/3297280.3297443

[46] P. Lipton, "Contrastive Explanation," *Royal Institute of Philosophy Supplement*, vol. 27, pp. 247–266, 2010. [Online]. Available: https://www.cambridge.org/core/product/identifier/S1358246100005130/type/journal_article

[47] J. Woodward and L. Ross, "Scientific Explanation," in *The Stanford Encyclopedia of Philosophy*, summer 2021 ed., E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2021. [Online]. Available: https://plato.stanford.edu/archives/sum2021/entries/scientific-explanation/

[48] C. Hitchcock, "Causal Models," in *The Stanford Encyclopedia of Philosophy*, spring 2023 ed., E. N. Zalta and U. Nodelman, Eds. Metaphysics Research Lab, Stanford University, 2023. [Online]. Available: https://plato.stanford.edu/archives/spr2023/entries/causal-models/

[49] Google Cloud, "AI Explainability Whitepaper," 2019.

[50] K. Sokol and P. Flach, "Counterfactual Explanations of Machine Learning Predictions: Opportunities and Challenges for AI Safety," *Proceedings of the AAAI Workshop on Artificial Intelligence Safety 2019*, Jan. 2019. [Online]. Available: http://ceur-ws.org/Vol-2301/paper_20.pdf

[51] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive Representation Learning: A Framework and Review," *IEEE Access*, vol. 8, pp. 193 907–193 934, 2020, conference Name: IEEE Access.

[52] ntrin, "ntrin/CIFAR_10_ml," Apr. 2021, original-date: 2020-09-26T02:40:15Z. [Online]. Available: https://github.com/ntrin/CIFAR_10_ML

[53] Ian H. Witten and Christopher J. Pal, "Imagenet Challenge - an overview | ScienceDirect Topics," 2017. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/imagenet-challenge

[54] W. Huang, M. Yi, X. Zhao, and Z. Jiang, "Towards the Generalization of Contrastive Self-Supervised Learning," Mar. 2023, arXiv:2111.00743 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2111.00743

[55] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," Jun. 2020, arXiv:2002.05709 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2002.05709

[56] C. M. Wild, "One Feature Attribution Method to (Supposedly) Rule Them All: Shapley Values," Jan. 2018. [Online]. Available: https://towardsdatascience.com/one-feature-attribution-method-to-supposedly-rule-them-all-shapley-values-f3e04534983d

[57] Credit One Bank, "Bank_loan_data," 2017. [Online]. Available: https://www.kaggle.com/dataforyou/bankloan

[58] https://www.facebook.com/baldthoughts, "Credit One Bank Review," Feb. 2023. [Online]. Available: https://www.investopedia.com/credit-one-bank-review-4842126

[59] Hadley Wickham, "Reshaping th data with the reshape package," 2007. [Online]. Available: URLhttp://www.jstatsoft.org/v21/i12/paper.

[60] IBM Cloud, "AI Explainability 360 (v0.2.1)," Apr. 2022, original-date: 2019-07-11T07:17:48Z. [Online]. Available: https://github.com/Trusted-AI/AIX360/blob/e06d186eaa67a9f42b183372933e9bbad64d8fd5/examples/tutorials/HELOC.ipynb

Appendices

Appendix A

Response from ChatGPT

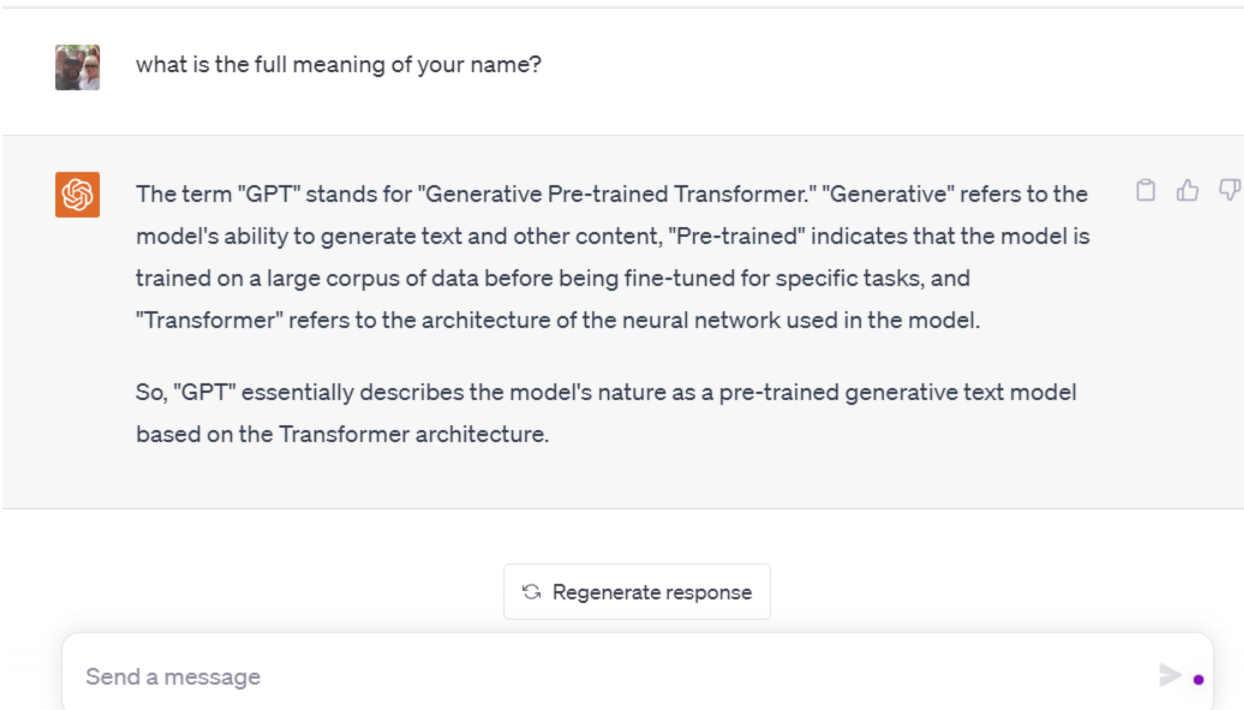


Figure A.1: *What is ChatGPT*

In this notebook I will preprocess a dataset for binary classification task. This will be done with a neural network. The specific task is to classify bank customers into possible loan-defaulters or non-defaulters

Appendix B

The data set is from the Credit One Bank and can be downloaded from Kaggle

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
pd.set_option('display.max_columns',None)
import matplotlib.pyplot as plt
import seaborn as sns
import os

import warnings
warnings.filterwarnings('ignore')

print('Imported!')
```

Imported!

In [2]:
```
os.listdir('data')
```

Out[2]:
```
['heloc_dataset_v1.csv',
 'Loan_Data for Classification.xls',
 'loan_df_prep.csv',
 'model_1lr',
 'Par_Data for Logistic Regression.xlsx']
```

In [3]:
```
# The link to the excel file containing the 2 sheets: Data and Sheet1

xls_link = 'data/Loan_Data for Classification.xls'
```

In [4]:
```
# Read in the excel file

xls = pd.ExcelFile(xls_link)
```

In [5]:
```
# Next, let's print the sheet names

print(xls.sheet_names)
```

['Data', 'Sheet1']

In [6]:
```
# The data set is in the sheet named Data

df = xls.parse('Data')

# Print the head of the DataFrame df
df.head()
```

Out[6]:

	Data to Analyse "Default on Payment"	Unnamed: 1	Unnamed: 2	Unnamed: 3	Unnamed: 4	Unnamed: 5	Unnamed: 6	Unnamed: 7	Unnamed: 8	Unnamed: 9	Unnamed: 10
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Customer_ID	Status_Checking_Acc	Duration_in_Months	Credit_History	Purposre_Credit_Taken	Credit_Amount	Savings_Acc	Years_At_Present_Employment	Inst_Rt_Income	Marital_Status_Gender	Other_Debtors_Guarantors
2	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
3	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
4	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101

The dataset was read wrong, with the header messed up. I shall correct that...

In [7]:
```
# First check the shape

df.shape
```

Out[7]: (5002, 23)

In [8]:
```
# reading the data with the header parameter specifying row 2,

df = xls.parse('Data', header=2)

# Print the head of the DataFrame df
df.head()
```

Out[8]:

	Customer_ID	Status_Checking_Acc	Duration_in_Months	Credit_History	Purposre_Credit_Taken	Credit_Amount	Savings_Acc	Years_At_Present_Employment	Inst_Rt_Income	Marital_Status_Gender	Other_Debtors_Guarantors
0	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
1	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
2	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101
3	100004	A11	42	A32	A42	7882	A61	A74	2	A93	A103
4	100005	A11	24	A33	A40	4870	A61	A73	3	A93	A101

About the Data

The data set is from The Credit One bank Credit One Bank is an online-only bank that exclusively offers credit cards. It is a technology and data-driven financial services company that is based in Las Vegas. The bank offers American Express and Visa credit cards to millions of customers across the United States.

This data is a real-life data that has been fully anonymized to protect the rights of customers. Yet, it holds real values that were used to train a classifier to predict if a customer/loan applicant is likely to default on re-paying their loan if advanced.

This data set contains the credit card application details of 5000 customers, spanning 22 different feature values or variables that offer insight to the personal and financial circumstances of the applicants. The data as well as it's data dictionary (A document showing the respective variables in the data set and their meanings) can be downloaded via this link on Kaggle

In [9]:
```
# Let's check the info and dtypes

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 23 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Customer_ID                  5000 non-null   int64
 1   Status_Checking_Acc          5000 non-null   object
 2   Duration_in_Months           5000 non-null   int64
 3   Credit_History               5000 non-null   object
 4   Purposre_Credit_Taken        5000 non-null   object
 5   Credit_Amount                5000 non-null   int64
 6   Savings_Acc                  5000 non-null   object
 7   Years_At_Present_Employment  5000 non-null   object
 8   Inst_Rt_Income               5000 non-null   int64
 9   Marital_Status_Gender        5000 non-null   object
 10  Other_Debtors_Guarantors     5000 non-null   object
 11  Current_Address_Yrs          5000 non-null   int64
 12  Property                     5000 non-null   object
 13  Age                          5000 non-null   int64
 14  Other_Inst_Plans             5000 non-null   object
 15  Housing                      5000 non-null   object
 16  Num_CC                       5000 non-null   int64
 17  Job                          5000 non-null   object
 18  Dependents                   5000 non-null   int64
 19  Telephone                    5000 non-null   object
 20  Foreign_Worker               5000 non-null   object
 21  Default_On_Payment           5000 non-null   int64
 22  Count                        5000 non-null   int64
dtypes: int64(10), object(13)
memory usage: 898.6+ KB
```

With each variable non-null, it means that there are no missing values

In [10]:
```
# Let's check the stats for columns of object dtype and transpose it for readability

df.describe(include='object').T
```

Out[10]:

	count	unique	top	freq
Status_Checking_Acc	5000	4	A14	1970
Credit_History	5000	5	A32	2660
Purposre_Credit_Taken	5000	10	A43	1400
Savings_Acc	5000	5	A61	3015
Years_At_Present_Employment	5000	5	A73	1695
Marital_Status_Gender	5000	4	A93	2740
Other_Debtors_Guarantors	5000	3	A101	4535
Property	5000	4	A123	1660
Other_Inst_Plans	5000	3	A143	4070
Housing	5000	3	A152	3565
Job	5000	4	A173	3150
Telephone	5000	2	A191	2980
Foreign_Worker	5000	2	A201	4815

In [11]:
```
# Now let's check the stats for numeric columns and transpose it for readability

df.describe().T
```

Out[11]:

	count	mean	std	min	25%	50%	75%	max
Customer_ID	5000.0	102500.500	1443.520003	100001.0	101250.75	102500.5	103750.25	105000.0
Duration_in_Months	5000.0	20.903	12.053988	4.0	12.00	18.0	24.00	72.0
Credit_Amount	5000.0	3271.258	2821.687329	250.0	1365.50	2319.5	3972.25	18424.0
Inst_Rt_Income	5000.0	2.973	1.118367	1.0	2.00	3.0	4.00	4.0
Current_Address_Yrs	5000.0	2.845	1.103276	1.0	2.00	3.0	4.00	4.0
Age	5000.0	35.546	11.370917	19.0	27.00	33.0	42.00	75.0
Num_CC	5000.0	1.407	0.577423	1.0	1.00	1.0	2.00	4.0
Dependents	5000.0	1.155	0.361941	1.0	1.00	1.0	1.00	2.0
Default_On_Payment	5000.0	0.299	0.457866	0.0	0.00	0.0	1.00	1.0
Count	5000.0	1.000	0.000000	1.0	1.00	1.0	1.00	1.0

We can see some interesting info here. For example, the average credit amount is about $3200, while the average age of loan applicants is 35.

In [12]:
```
# Let's see the head again

df.head()
```

Out[12]:

	Customer_ID	Status_Checking_Acc	Duration_in_Months	Credit_History	Purposre_Credit_Taken	Credit_Amount	Savings_Acc	Years_At_Present_Employment	Inst_Rt_Income	Marital_Status_Gender	Other_Debtors_Guarantors
0	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
1	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
2	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101
3	100004	A11	42	A32	A42	7882	A61	A74	2	A93	A103
4	100005	A11	24	A33	A40	4870	A61	A73	3	A93	A101

I also notice the count column seems to have only 1's in it and no reason why, let's consult the Data Dictionary to find out what it means and deal with it as we pre-process the Data

In [13]:
```
# Let's see the unique values in count...

df.Count.unique()
```

Out[13]: array([1], dtype=int64)

In [14]:
```
# with just ones within and no mention in the data dictionary, I will drop the Count column
# First make a copy of the data

loan_df = df.copy()

loan_df.head(3)
```

Out[14]:

	Customer_ID	Status_Checking_Acc	Duration_in_Months	Credit_History	Purposre_Credit_Taken	Credit_Amount	Savings_Acc	Years_At_Present_Employment	Inst_Rt_Income	Marital_Status_Gender	Other_Debtors_Guarantors
0	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
1	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
2	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101

In [15]:
```
# Now, drop the count column

loan_df.drop(columns=['Count'], inplace=True)
```

In [16]:
```
# Let's see

loan_df.head(3)
```

Out[16]:

	Customer_ID	Status_Checking_Acc	Duration_in_Months	Credit_History	Purposre_Credit_Taken	Credit_Amount	Savings_Acc	Years_At_Present_Employment	Inst_Rt_Income	Marital_Status_Gender	Other_Debtors_Guarantors
0	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
1	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
2	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101

Data Preprocessing

In [17]:
```
# Checking for Duplicate data:

loan_df.duplicated().sum()
```

Out[17]: 0

Cleaning Column Names:

I can see we need to clean some of the column names for empty spaces or misspelt words. Also, it's better for all to be lower case, not camel case Let's do so now.

In [18]:
```
list(loan_df)
```

Out[18]:
```
['Customer_ID',
 'Status_Checking_Acc',
 'Duration_in_Months',
 'Credit_History',
 'Purposre_Credit_Taken',
 'Credit_Amount',
 'Savings_Acc',
 'Years_At_Present_Employment',
 'Inst_Rt_Income',
 'Marital_Status_Gender',
 'Other_Debtors_Guarantors',
 'Current_Address_Yrs',
 'Property',
 'Age',
 'Other_Inst_Plans ',
 'Housing',
 'Num_CC',
 'Job',
 'Dependents',
 'Telephone',
 'Foreign_Worker',
 'Default_On_Payment']
```

In [19]:
```
# Lowering and removing trailing white spaces

new_col = [x.lower().strip() for x in list(loan_df)]

# Let's see the new columns
new_col
```

Out[19]:
```
['customer_id',
 'status_checking_acc',
 'duration_in_months',
 'credit_history',
 'purposre_credit_taken',
 'credit_amount',
 'savings_acc',
 'years_at_present_employment',
 'inst_rt_income',
 'marital_status_gender',
 'other_debtors_guarantors',
 'current_address_yrs',
 'property',
 'age',
 'other_inst_plans',
 'housing',
 'num_cc',
 'job',
 'dependents',
 'telephone',
 'foreign_worker',
 'default_on_payment']
```

In [20]:
```
# Let's update the columns

loan_df.columns = new_col

loan_df.head(3)
```

Out[20]:

	customer_id	status_checking_acc	duration_in_months	credit_history	purposre_credit_taken	credit_amount	savings_acc	years_at_present_employment	inst_rt_income	marital_status_gender	other_debtors_guarantors
0	100001	A11	6	A34	A43	1169	A65	A75	4	A93	A101
1	100002	A12	48	A32	A43	5951	A61	A73	2	A92	A101
2	100003	A14	12	A34	A46	2096	A61	A74	2	A93	A101

In [21]:
```
# now let's rename some of the columns to more intuitive names...

loan_df.rename(
    columns={'purposre_credit_taken':'credit_purpose',
             'years_at_present_employment': 'yrs_present_employment',
             'inst_rt_income': 'inst_rate_income',
             'num_cc': 'num_curr_credits'
             },
    inplace=True
)
```

In [22]:
```
# Let's see the renamed columns

list(loan_df)
```

Out[22]:
```
['customer_id',
 'status_checking_acc',
 'duration_in_months',
 'credit_history',
 'credit_purpose',
 'credit_amount',
 'savings_acc',
 'yrs_present_employment',
 'inst_rate_income',
 'marital_status_gender',
 'other_debtors_guarantors',
 'current_address_yrs',
 'property',
 'age',
 'other_inst_plans',
 'housing',
 'num_curr_credits',
 'job',
 'dependents',
 'telephone',
 'foreign_worker',
 'default_on_payment']
```

Contiguous Column Groupings:

The variables in the dataset can be grouped into 3 broad classes:

1. customer_data,
2. job_and_personal_finance,
3. loan_credit_data

In [23]:
```
customer_data = ['customer_id', 'marital_status_gender', 'age', 'dependents']
job_and_personal_finance = ['status_checking_acc', 'savings_acc', 'yrs_present_employment', 'property',
                            'housing', 'job', 'telephone', 'foreign_worker', 'current_address_yrs']
loan_data = ['duration_in_months', 'credit_history', 'credit_purpose', 'credit_amount',
             'inst_rate_income', 'other_debtors_guarantors', 'other_inst_plans', 'num_curr_credits',
             'default_on_payment']
```

In [24]:
```
# Adding the 3 lists together

contiguous_cols = customer_data + job_and_personal_finance + loan_data
```

In [25]:
```
# confirming we didn't miss anything

set(contiguous_cols) - set(list(loan_df))
```

Out[25]: set()

The empty set above indicates both lists are identical... So I can go ahead and rearrange the columns

In [26]:
```
loan_df = loan_df[contiguous_cols]

# Let's see the 5 top rows...

loan_df.head()
```

Out[26]:

	customer_id	marital_status_gender	age	dependents	status_checking_acc	savings_acc	yrs_present_employment	property	housing	job	telephone	foreign_worker	current_address_yrs	duration_in_months	credit_history
0	100001	A93	67	1	A11	A65	A75	A121	A152	A173	A192	A201	4	6	A34
1	100002	A92	22	1	A12	A61	A73	A121	A152	A173	A191	A201	2	48	A32
2	100003	A93	49	2	A14	A61	A74	A121	A152	A172	A191	A201	3	12	A34
3	100004	A93	45	2	A11	A61	A74	A122	A153	A173	A191	A201	4	42	A32
4	100005	A93	53	2	A11	A61	A73	A124	A153	A173	A191	A201	4	24	A33

The loan_df columns are ordered in a logical fashion, with related columns together, forming tables.... let's start feature binarization and pre-processing

It looks like the customerID column may not have any predictive value, let's consult the data dictionary as it may need to be deleted... Having checked the data dictionary, I can delete it.

In [27]:
```
loan_df.drop(columns=['customer_id'], inplace=True)
```

Visualize The relationships between categorical variables and target variable

Dealing with categorical variables can be a little tricky. For example categorical columns like _Status_CheckingAcc_, _CreditHistory_, have codes representing some values. For example, in the _CreditHistory_ column, code A34 represents critical account/other credits, while code A32 represents existing credits paid back duly till now.

In [28]:
```
# Splitting categorical & numerical columns
cat_cols = loan_df.select_dtypes(['object'])
```

save the model weights

In [65]:
```
model.save_weights('loan_classifier_model', overwrite=True, save_format='h5')
print('loan_classifier saved!')
```

loan_classifier saved!

Evaluate the Model:

In [66]:
```
# let's make a prediction first on the test set...

pred = model.predict(x_test)

# round it up

pred = np.round(pred)
print(pred.shape)
```

(500, 1)

In [67]:
```
# Let's now score the model on the test set

score = model.evaluate(x_test, y_test, verbose=0)
score
```

Out[67]: [0.1339672408103943, 5.0, 16.0, 340.0, 145.0, 0.9354839, 0.90066344]

In [68]:
```
model.metrics_names
```

Out[68]: ['loss', 'fn', 'fp', 'tn', 'tp', 'precision', 'recall']

In [69]:
```
for name, val in zip(model.metrics_names, score):
    print(name+': ', round(val,3))
```

```
loss: 0.13
fn: 5.0
fp: 16.0
tn: 340.0
tp: 145.0
precision: 0.94
recall: 0.97
```

So we can see the model's performance above on the test set, for Precision and Recall metrics. Quite impressive.

Compute Contrastive Explanations for a few customers

In [70]:
```
from aix360.algorithms.contrastive import CEMExplainer, KerasClassifier
print('Done!')
```

Using TensorFlow backend.
Done!

In [71]:
```
class_names = ['non_default', 'default']
```

Given the trained NN model to decide on loan defaulters, let us first examine an applicant who was classified as a defaulter to both the model prediction and the test set. Then compute what (minimal) changes to his/her application would lead to approval (i.e. finding pertinent negatives). We will then look at another applicant who was classified as a non-defaulter and ascertain features that would minimally suffice in forming a positive outcome (i.e. finding pertinent positives).

In [72]:
```
# creating a dataframe from the model predictions

pred_df = pd.DataFrame(x_test)
pred_df.columns = list(loan_df_dummies.columns[:-1])
pred_df['prediction'] = pred
```

In [73]:
```
pred_df.head(3)
```

Out[73]:

	age	dependents	current_address_yrs	duration_in_months	credit_amount	inst_rate_income	num_curr_credits	marital_status_gender_A93	marital_status_gender_A94	status_checking_acc_A13	status_checking_acc_A14
0	-0.142714	-0.5	0.500000	-0.394116	-0.427479	0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5
1	0.355571	-0.5	-0.166667	-0.431765	-0.185040	-0.500000	-0.166667	-0.5	-0.5	-0.5	-0.5
2	0.355571	-0.5	0.166667	-0.470588	-0.472323	-0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5

In [74]:
```
# creating a dataframe from the test set

test_df = pd.DataFrame(x_test)
test_df.columns = list(loan_df_dummies.columns[:-1])
test_df['target'] = y_test
```

In [75]:
```
test_df.head(3)
```

Out[75]:

	age	dependents	current_address_yrs	duration_in_months	credit_amount	inst_rate_income	num_curr_credits	marital_status_gender_A93	marital_status_gender_A94	status_checking_acc_A13	status_checking_acc_A14
0	-0.142714	-0.5	0.500000	-0.394116	-0.427479	0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5
1	0.355571	-0.5	-0.166667	-0.431765	-0.185040	-0.500000	-0.166667	-0.5	-0.5	-0.5	-0.5
2	0.355571	-0.5	0.166667	-0.470588	-0.472323	-0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5

Next, let's randomly pick a set of 3 customers who are classed as defaulters in both the prediction and test set... Then we will do the same for non-defaulters

In [76]:
```
def target_selector(val, n=30):
    """Get a sample of random IDs
    based on target value from test set
    return: DataFrame
    """
    select = test_df[test_df.target == val]

    # getting a random distribution of 30 customers
    select = select.sample(n, random_state=seed)
    return select
```

In [77]:
```
# selecting only defaulters
defs = target_selector(1)

defs.head(3)
```

Out[77]:

	age	dependents	current_address_yrs	duration_in_months	credit_amount	inst_rate_income	num_curr_credits	marital_status_gender_A93	marital_status_gender_A94	status_checking_acc_A13	status_checking_acc_A14
323	-0.142714	-0.5	0.500000	-0.394116	-0.427479	0.500000	-0.166667	-0.5	-0.5	-0.5	-0.5
384	-0.339286	-0.5	0.500000	-0.235235	-0.389608	0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5
91	-0.321429	-0.5	-0.166667	-0.392353	-0.315229	0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5

In [78]:
```
# selecting 3 random defaulters

def_ids = []

for ind, val in zip(defs.index, defs.target):
    if pred_df.at[ind, 'prediction'] == val:
        def_ids.append(ind)
    if len(def_ids) == 3:
        break
def_ids
```

Out[78]: [323, 384, 91]

In [79]:
```
# selecting only non-defaulters
ndefs = target_selector(0)

ndefs.head(3)
```

Out[79]:

	age	dependents	current_address_yrs	duration_in_months	credit_amount	inst_rate_income	num_curr_credits	marital_status_gender_A93	marital_status_gender_A94	status_checking_acc_A13	status_checking_acc_A14
57	0.133929	-0.5	0.500000	-0.470588	-0.478780	-0.500000	-0.166667	-0.5	-0.5	-0.5	-0.5
429	-0.375000	-0.5	-0.166667	-0.372529	-0.361671	-0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5
183	-0.571429	-0.5	0.166667	-0.294118	-0.375271	0.166667	-0.166667	-0.5	-0.5	-0.5	-0.5

In [80]:
```
# selecting 3 non-defaulting customers

ndef_ids = []

for ind, val in zip(ndefs.index, ndefs.target):
    if pred_df.at[ind, 'prediction'] == val:
        ndef_ids.append(ind)
    if len(ndef_ids) == 3:
        break
ndef_ids
```

Out[80]: [57, 429, 183]

So we have two lists, each containing 3 random defaulting and non-defaulting customers corresponding to the predicted and target classes.

Computing PN for the first defaulter...

In [81]:
```
idx = def_ids[0]

X = x_test[idx].reshape((1,) + x_test[idx].shape)
print("Computing PN for Sample:", idx)
print("Prediction made by the model:", class_names[np.argmax(model.predict_proba(X))])
print("Prediction probabilities:", model.predict_proba(X))
print("")

mymodel = KerasClassifier(model)
explainer = CEMExplainer(mymodel)

arg_mode = 'PN' # Find pertinent negatives
arg_max_iter = 1000 # Maximum number of iterations to search for the optimal PN for given parameter settings
arg_init_const = 10.0 # Initial coefficient value for main loss term that encourages class change
arg_b = 9 # No. of updates to the coefficient of the main loss term
arg_kappa = 0.2 # Minimum confidence gap between the PNs (changed) class probability and original class' probability
arg_beta = 1e-1 # Controls sparsity of the solution (L1 loss)
arg_gamma = 100 # Controls how much to adhere to a (optionally trained) auto-encoder
my_AE_model = None # Pointer to an auto-encoder
arg_alpha = 0.01 # Penalizes L2 norm of the solution
arg_threshold = 1. # Automatically turn off features <= arg_threshold if arg_threshold < 1
arg_offset = 0.5 # the model assumes classifier trained on data normalized
                 # in [-arg_offset, arg_offset] range, where arg_offset is 0 or 0.5
# Find PN for applicant 1
(adv_pn, delta_pn, info_pn) = explainer.explain_instance(X, arg_mode, my_AE_model, arg_kappa, arg_b,
                                                         arg_max_iter, arg_init_const, arg_beta, arg_gamma,
                                                         arg_alpha, arg_threshold, arg_offset)
```

```
Computing PN for Sample: 323
Prediction made by the model: [[1.]]
Prediction probabilities: non_default

WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:60: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:151: The name tf.assign is deprecated. Please use tf.compat.v1.assign instead.

WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:181: The name tf.train.polynomial_decay is deprecated. Please use tf.compat.v1.train.polynomial_decay instead.

WARNING:tensorflow:From C:\Users\...\site-packages\tensorflow_core\python\keras\optimizer_v2\learning_rate_schedule.py:409: div (from tensorflow.python.ops.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Deprecated in favor of operator or tf.math.divide.
WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:216: The name tf.train.GradientDescentOptimizer is deprecated. Please use tf.compat.v1.train.GradientDescentOptimizer instead.

WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:217: The name tf.global_variables is deprecated. Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From C:\Users\...\aix360\algorithms\contrastive\CEM_aen.py:230: The name tf.variables_initializer is deprecated. Please use tf.compat.v1.variables_initializer instead.

iter:0 const:[10.]
Loss_Overall:100012.0000, Loss_Attack:100012.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[10.]
Loss_Overall:100012.0000, Loss_Attack:100012.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[100.]
Loss_Overall:1000120.0000, Loss_Attack:1000120.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[100.]
Loss_Overall:1000120.0000, Loss_Attack:1000120.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[1000.]
Loss_Overall:10001200.0000, Loss_Attack:10001200.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[1000.]
Loss_Overall:10001200.0000, Loss_Attack:10001200.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[10000.]
Loss_Overall:100012000.0000, Loss_Attack:100012000.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[10000.]
Loss_Overall:100012000.0000, Loss_Attack:100012000.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[100000.]
Loss_Overall:1000120000.0000, Loss_Attack:1000120000.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[100000.]
Loss_Overall:1000120000.0000, Loss_Attack:1000120000.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[1000000.]
Loss_Overall:10001199232.0000, Loss_Attack:10001199232.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[1000000.]
Loss_Overall:10001199232.0000, Loss_Attack:10001199232.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[10000000.]
Loss_Overall:100011999232.0000, Loss_Attack:100011999232.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[10000000.]
Loss_Overall:100011999232.0000, Loss_Attack:100011999232.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[1.e+08]
Loss_Overall:1000119992320.0000, Loss_Attack:1000119992320.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[1.e+08]
Loss_Overall:1000119992320.0000, Loss_Attack:1000119992320.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:0 const:[1.e+09]
Loss_Overall:10001200447488.0000, Loss_Attack:10001200447488.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000

iter:500 const:[1.e+09]
Loss_Overall:10001200447488.0000, Loss_Attack:10001200447488.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:1.0000, max_nontarget_lab_score:-10000.0000
```

Let us start by examining one particular defaulter application that was denied for the first applicant. We showcase below how the decision could have been different through minimal changes to the profile conveyed by the pertinent negative. We also indicate the importance of different features to produce the change in the application status. The column delta in the table below indicates the necessary deviations for each of the features to produce this change. A human friendly explanation is then provided based on these deviations following the feature importance plot.

In [82]:
```
Xpn = adv_pn
classes = [ class_names[np.argmax(model.predict_proba(X))], class_names[np.argmax(model.predict_proba(Xpn))], 'NIL' ]

print("Sample:", idx)
print("prediction(X)", model.predict_proba(X), class_names[np.argmax(model.predict_proba(X))])
print("prediction(Xpn)", model.predict_proba(Xpn), class_names[np.argmax(model.predict_proba(Xpn))] )

X_re = rescale(X) # Convert values back to original scale from normalized
Xpn_re = rescale(Xpn)
Xpn_re = np.around(Xpn_re.astype(np.double), 2)

delta_re = Xpn_re - X_re
delta_re = np.around(delta_re.astype(np.double), 2)
delta_re[np.absolute(delta_re) < 1e-4] = 0

X3 = np.vstack((X_re, Xpn_re, delta_re))

dfre = pd.DataFrame.from_records(X3) # Create dataframe to display original point, PN and difference (delta)
dfre[X.shape[1]] = classes

dfre.columns = loan_df_dummies.columns
dfre.rename(index={0:'X',1:'X_PN', 2:'(X_PN - X)'}, inplace=True)
dfret = dfre.transpose()


def highlight_ce(s, col, ncols):
    if (type(s[col]) != str):
        if (s[col] > 0):
            return(['background-color: yellow']*ncols)
    return(['background-color: white']*ncols)

dfret.style.apply(highlight_ce, col='(X_PN - X)', ncols=3, axis=1)
```

```
Sample: 323
prediction(X) [[1.]] non_default
prediction(Xpn) [[0.9961134]] non_default
```

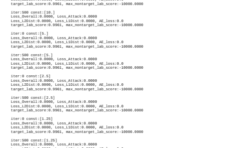
Out[82]:

	X	X_PN	(X_PN - X)
age	39.000000	19.000000	-20.000000
dependents	2.000000	1.000000	-1.000000
current_address_yrs	3.000000	1.000000	-2.000000
duration_in_months	24.000000	4.000000	-20.000000
credit_amount	946.999756	250.000000	-697.000000
inst_rate_income	4.000000	1.000000	-3.000000
num_curr_credits	1.000000	1.000000	0.000000
marital_status_gender_A93	1.000000	0.000000	-1.000000
marital_status_gender_A94	0.000000	0.000000	0.000000
status_checking_acc_A13	0.000000	0.000000	0.000000
status_checking_acc_A14	0.000000	0.000000	0.000000
savings_acc_A63	0.000000	0.000000	0.000000
savings_acc_A64	0.000000	0.000000	0.000000
savings_acc_A65	0.000000	0.000000	0.000000
yrs_present_employment_A74	1.000000	0.000000	-1.000000
yrs_present_employment_A75	0.000000	0.000000	0.000000
property_A121	0.000000	0.000000	0.000000
housing_A152	0.000000	0.000000	0.000000
job_A172	0.000000	0.000000	0.000000
job_A173	1.000000	0.000000	-1.000000
telephone_A192	0.000000	0.000000	0.000000
foreign_worker_A202	0.000000	0.000000	0.000000
credit_history_A34	0.000000	0.000000	0.000000
credit_purpose_A41	0.000000	0.000000	0.000000
credit_purpose_A43	0.000000	0.000000	0.000000
credit_purpose_A48	0.000000	0.000000	0.000000
other_debtors_guarantors_A101	1.000000	0.000000	-1.000000
other_debtors_guarantors_A103	0.000000	0.000000	0.000000
other_inst_plans_A143	0.000000	0.000000	0.000000
default_on_payment	non_default	non_default	NIL

In [83]:
```
plt.rcdefaults()
fi = abs((X-Xpn).astype('double'))/np.std(train_features.astype('double'), axis=0) # Compute PN feature importance
objects = loan_df_dummies.columns[-2::-1]
y_pos = np.arange(len(objects))
performance = fi[0, -1::-1]

plt.barh(y_pos, performance, align='center', alpha=0.5) # bar chart
plt.yticks(y_pos, objects) # Display features on y-axis
plt.xlabel('weight') # x-label
plt.title('PN (feature importance)') # Heading

plt.show() # Display PN feature importance
```



Computing Pertinent Positives

In [84]:
```
# Some interesting user samples to try: 9 11 24
idx = ndef_idx[0]

X = x_test[idx].reshape((1,) + x_test[idx].shape)
print("Computing PP for Sample:", idx)
print("Prediction made by the model:", class_names[np.argmax(model.predict_proba(X))])
print("Prediction probabilities:", model.predict_proba(X))
print("")

mymodel = KerasClassifier(model)
explainer = CEMExplainer(mymodel)

arg_mode = 'PP' # Find pertinent positives
arg_max_iter = 1000 # Maximum number of iterations to search for the optimal PN for given parameter settings
arg_init_const = 10.0 # Initial coefficient value for main loss term that encourages class change
arg_b = 9 # No. of updates to the coefficient of the main loss term
arg_kappa = 0.2 # Minimum confidence gap between the PNs (changed) class probability and original class' probability
arg_beta = 10.0 # Controls sparsity of the solution (L1 loss)
arg_gamma = 100 # Controls how much to adhere to a (optionally trained) auto-encoder
my_AE_model = None # Pointer to an auto-encoder
arg_alpha = 0.1 # Penalizes L2 norm of the solution
arg_threshold = 0.0 # Automatically turn off features <= arg_threshold if arg_threshold < 1
arg_offset = 0.5 # the model assumes classifier trained on data normalized
                 # in [-arg_offset, arg_offset] range, where arg_offset is 0 or 0.5
(adv_pp, delta_pp, info_pp) = explainer.explain_instance(X, arg_mode, my_AE_model, arg_kappa, arg_b,
                                                         arg_max_iter, arg_init_const, arg_beta, arg_gamma,
                                                         arg_alpha, arg_threshold, arg_offset)
```

```
Computing PP for Sample: 57
Prediction made by the model: non_default
Prediction probabilities: [[0.18547778e-07]]

iter:0 const:[10.]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[10.]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[5.]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[5.]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[2.5]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[2.5]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[1.25]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[1.25]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[0.625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[0.625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[0.3125]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[0.3125]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[0.15625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[0.15625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[0.078125]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[0.078125]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:0 const:[0.0390625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000

iter:500 const:[0.0390625]
Loss_Overall:0.0000, Loss_Attack:0.0000
Loss_L2Dist:0.0000, Loss_L1Dist:0.0000, AE_loss:0.0
target_lab_score:0.9961, max_nontarget_lab_score:-10000.0000
```

In [85]:
```
Xpp = delta_pp
classes = [ class_names[np.argmax(model.predict_proba(X))], class_names[np.argmax(model.predict_proba(Xpp))]]

print("PP for Sample:", idx)
print("Prediction(Xpp) :", class_names[np.argmax(model.predict_proba(Xpp))])
print("Prediction probabilities for Xpp:", model.predict_proba(Xpp))
print("")

X_re = rescale(X) # Convert values back to original scale from normalized
adv_pp_re = rescale(adv_pp)
#Xpp_re = X_re - adv_pp_re
Xpp_re = rescale(Xpp)
Xpp_re = np.around(Xpp_re.astype(np.double), 2)
Xpp_re[Xpp_re < 1e-4] = 0

X2 = np.vstack((X_re, Xpp_re))

dfpp = pd.DataFrame.from_records(X2.astype('double')) # Showcase a dataframe for the original point and PP
dfpp[X.shape[1]] = classes
dfpp.columns = loan_df_dummies.columns
dfpp.rename(index={0:'X',1:'X_PP'}, inplace=True)
dfppt = dfpp.transpose()

dfppt.style.apply(highlight_ce, col='X_PP', ncols=2, axis=1)
```

```
PP for Sample: 57
Prediction(Xpp) : non_default
Prediction probabilities for Xpp: [[0.9961134]]
```

Out[85]:

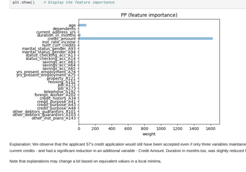
	X	X_PP
age	54.000000	19.000000
dependents	1.000000	1.000000
current_address_yrs	4.000000	1.000000
duration_in_months	6.000000	4.000000
credit_amount	971.999756	250.000000
inst_rate_income	1.000000	1.000000
num_curr_credits	1.000000	1.000000
marital_status_gender_A93	0.000000	0.000000
marital_status_gender_A94	0.000000	0.000000
status_checking_acc_A13	0.000000	0.000000
status_checking_acc_A14	1.000000	0.000000
savings_acc_A63	0.000000	0.000000
savings_acc_A64	0.000000	0.000000
savings_acc_A65	0.000000	0.000000
yrs_present_employment_A74	0.000000	0.000000
yrs_present_employment_A75	1.000000	0.000000
property_A121	1.000000	0.000000
housing_A152	1.000000	0.000000
job_A172	0.000000	0.000000
job_A173	1.000000	0.000000
telephone_A192	1.000000	0.000000
foreign_worker_A202	0.000000	0.000000
credit_history_A34	0.000000	0.000000
credit_purpose_A41	0.000000	0.000000
credit_purpose_A43	0.000000	0.000000
credit_purpose_A48	0.000000	0.000000
other_debtors_guarantors_A101	1.000000	0.000000
other_debtors_guarantors_A103	0.000000	0.000000
other_inst_plans_A143	1.000000	0.000000
default_on_payment	non_default	non_default

In [86]:
```
plt.rcdefaults()
fi = abs(Xpp_re.astype('double'))/np.std(x_train.astype('double'), axis=0) # Compute PP feature importance

objects = loan_df_dummies.columns[-2::-1]
y_pos = np.arange(len(objects)) # Get input feature names
performance = fi[0, -1::-1]

plt.barh(y_pos, performance, align='center', alpha=0.5) # Bar chart
plt.yticks(y_pos, objects) # Plot feature names on y-axis
plt.xlabel('weight') #x-label
plt.title('PP (feature importance)') # Figure heading

plt.savefig('img/pp_fi')
plt.show()    # Display the feature importance
```



Explanation: We observe that the applicant 57's credit application would still have been accepted even if only three variables maintained their original level - the Dependents, the Installment-rate-Income, and the number of current credits - and had a significant reduction in an additional variable - Credit Amount. Duration in months too, was slightly reduced from 6 to 4.

Note that explanations may change a bit based on equivalent values in a local minima.