Degree Project in Engineering Physics and Mathematics

Second cycle, 30 credits

Applying Machine Learning to Number-Theoretical Data

A Focus on Class Groups of Imaginary Quadratic Fields

HUGO STRÄNG

Applying Machine Learning to Number-Theoretical Data

A Focus on Class Groups of Imaginary Quadratic Fields

HUGO STRÄNG

Master's Programme, Mathematics, 120 credits
Date: August 1, 2024

Supervisor: Pär Kurlberg
Examiner: Pär Kurlberg
School of Engineering Sciences
Swedish title: Tillämpning av maskininlärning på talteoretisk data
Swedish subtitle: Fokus på klassgrupper av imaginära kvadratiska talkroppar

Abstract

This report considers problems from number theory through the lens of machine learning. The work encompasses class groups of imaginary quadratic fields and places particular focus on their orders and discriminants. The class group of discriminant $d < 0$ is denoted by $H(d)$ and its order — also referred to as the class number — is denoted by $h(d)$. Methods for predicting divisibility properties of class numbers based on discriminant information are investigated. Moreover, we implement methods for approximating the number $\mathcal{F}(h)$ of negative fundamental prime discriminants with class groups of order h. Class groups have been studied since the early 19th century and extensive theory related to these objects has been developed over the years. Yet, a large part of this theory consists of conjectures supported by heuristics, and many related questions remain unanswered. The purpose of this thesis is to explore whether methods from machine learning can shed new light on this domain. We find that our implemented models reach relatively low errors when predicting $\mathcal{F}(h)$ based on h. However, this only works when the models have knowledge about certain divisibility properties of h. Furthermore, we conclude that the methods reviewed in this thesis fails when it comes to separating discriminants based on three and five divisibility of their corresponding class numbers. Higher accuracy is reported in regards to separating discriminants, within certain feature spaces, based on the number of prime factors in their corresponding class numbers. However, these results can largely be explained by already established theory.

Keywords

Class group, Class number, Deep learning, Fundamental discriminant, Machine learning, Number theory, Quadratic number field

Sammanfattning

Denna rapport behandlar problem från talteori utifrån ett maskininlärningsperspektiv. Arbetet omfattar klassgrupper av imaginära kvadratiska talkroppar och fokuserar särskilt på deras ordningar och diskriminanter. Klassgruppen med diskriminant $d < 0$ betecknas med $H(d)$ och dess ordning — även kallad klasstalet — betecknas med $h(d)$. Metoder för att förutsäga delbarhetsegenskaper hos klasstalet baserat på diskriminantinformation undersöks. Dessutom implementerar vi metoder för att approximera antalet $\mathcal{F}(h)$ negativa fundamentala primtalsdiskriminanter med klassgrupp av ordning h. Klassgrupper har studerats sedan början av 1800-talet och omfattande teori relaterad till dessa objekt har utvecklats genom åren. Ändå består en stor del av denna teori av förmodanden stödda av heuristik, och många relaterade frågor är obesvarade. Syftet med denna rapport är att undersöka om metoder från maskininlärning kan ge ny insikt inom detta område. Vi finner att våra implementerade modeller når relativt låga fel när de förutsäger $\mathcal{F}(h)$ baserat på h. Dock fungerar detta endast när modellerna har kunskap om vissa delbarhetsegenskaper hos h. Vidare drar vi slutsatsen att de metoder som granskas i detta arbete misslyckas när det gäller att separera diskriminanter baserat på tre- och femdelbarhet hos deras motsvarande klasstal. Högre noggrannhet rapporteras när det gäller att separera diskriminanter, inom vissa representationsrum, baserat på antalet primtalsfaktorer i deras motsvarande klasstal. Dock kan dessa resultat till stor del förklaras av redan etablerad teori.

Nyckelord

Klassgrupp, Klasstal, Djupinlärning, Fundamental diskriminant, Maskininlärning, Talteori, Kvadratisk talkropp

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Professor Pär Kurlberg, for the opportunity to explore such an exciting research area. His guidance has been essential to the fulfilment of this work.

I am also thankful to my friends, Joel Wållberg and Hugo Fernström, for taking the time to read and give feedback on the text. Their insightful comments and questions have significantly enhanced the following report. I extend my thanks to all my friends who have shown support.

I would like to thank my girlfriend, Angela, for her steady encouragement throughout this work.

Finally, I am deeply grateful to my family — their support has been instrumental to the completion of this master thesis.

Stockholm, August 2024
Hugo Sträng

Contents

1 Introduction **1**
1.1 Preliminaries . . . 1
1.2 Background . . . 5
1.3 Research Questions . . . 8
1.4 Structure of the Thesis . . . 8

2 Related Work **9**
2.1 Class Group Heuristics . . . 9
2.2 Machine Learning in Number Theory . . . 11

3 Methodology **15**
3.1 Predicting $\mathcal{F}(h)$ . . . 16
3.1.1 Data Collection . . . 16
3.1.2 Problem Setup and Model Design . . . 16
3.1.3 Benchmarks . . . 19
3.1.4 Experiments . . . 20
3.2 Predicting Class Numbers . . . 22
3.2.1 Data Collection . . . 22
3.2.2 Problem Setup and Model Design . . . 22
3.2.3 Benchmarks . . . 24
3.2.4 Experiments . . . 25

4 Results **27**
4.1 Predicting $\mathcal{F}(h)$ . . . 27
4.2 Predicting Class Numbers . . . 27

5 Discussion **47**
5.1 Predicting $\mathcal{F}(h)$ . . . 47
5.2 Predicting Class Numbers . . . 48

6 Conclusions and Future Work **53**
6.1 Conclusions 53
6.2 Limitations 53
6.3 Future Work 54
6.4 Reflections 54

References **57**

A The Form Class Group **61**

B Computing Coefficients **63**

C Supplementary Figures **69**

Chapter 1

Introduction

Over the past few decades, the field of artificial intelligence (AI) has experienced unprecedented growth. This form of machine intelligence has achieved impressive feats to say the least: from Deep Blue's victory against Garry Kasparov in 1997 to Google DeepMind's recent AlphaGeometry [1] — an AI system capable of solving IMO-level geometry problems.

Machine learning (ML) and deep learning (DL) are subsets of AI that focuses on enabling machines to learn from data. Due to the advancement of big data and hardware, methods from these areas have achieved outstanding success in a variety of applications, including natural language processing, speech recognition and image classification. [2]

In regards to pure mathematics, the composition of computers, big data and AI enables exploration in novel ways. Leveraging AIs abilities in discovering patterns from data could expedite advances in experimental math. For instance, these tools can aid in formulating and supporting conjectures that can eventually be proven. [3]

The purpose of this paper is to explore whether machine learning methods can perform well in certain tasks from computational number theory. In particular, we consider problems regarding class groups of imaginary quadratic fields.

1.1 Preliminaries

Defining the ideal class group of an imaginary quadratic field or, more generally, a number field requires a few underlying definitions.

Definition 1.1.1. Let K be an extension field of $\mathbb{Q}$. If K, considered as a $\mathbb{Q}$-vector space, has finite dimension, we say that K is a *number field*. The

dimension of K, as a $\mathbb{Q}$-vector space, is called the *degree* of K, which we denote as $\deg(K)$.

In other words, a number field is a field that contains $\mathbb{Q}$ as a subfield and is finite-dimensional considered as a vector space over $\mathbb{Q}$. For example, the Gaussian rationals, $\mathbb{Q}(i) = \{a + bi : a, b \in \mathbb{Q}\}$, is a number field with degree 2. In fact, the Gaussian rationals is an example of an imaginary quadratic field.

Definition 1.1.2. Let K be a number field of the form $\mathbb{Q}(\sqrt{d}) = \{a + b\sqrt{d} : a, b \in \mathbb{Q}\}$ where $d \notin \{0, 1\}$ is a square-free integer. Then K is a *quadratic field*. If $d > 0$ we say that K is a *real quadratic field* and if $d < 0$ we say that K is an *imaginary quadratic field*.

We need the definition of an algebraic integer.

Definition 1.1.3. Consider $\alpha \in \mathbb{C}$. If there exists a monic polynomial (a polynomial whose leading coefficient is 1) $p(x) \in \mathbb{Z}[x]$ with $p(\alpha) = 0$, then we say that α is an *algebraic integer*. We denote the set of all algebraic integers by $\overline{\mathbb{Z}}$.

Thus, $3i \in \overline{\mathbb{Z}}$ since it is a root of $p(x) = x^2 + 9 \in \mathbb{Z}[x]$. In fact, $\overline{\mathbb{Z}}$ is a subring of $\mathbb{C}$. Thus it follows that the intersection between the algebraic integers and any number field is a ring.

Definition 1.1.4. If K is a number field, then the ring $\mathbb{Z}_K := K \cap \overline{\mathbb{Z}}$ is called *the ring of integers of* K.

At this point we need to recall the definition of a module over a ring.

Definition 1.1.5. A *module* over a ring $\mathscr{R}$ is an abelian group $(M, +)$ together with a map $\cdot : \mathscr{R} \times M \to M$ such that $\forall a, b \in \mathscr{R}$ and $x, y \in M$,
(i) $a \cdot (x + y) = a \cdot x + a \cdot y$,
(ii) $(a + b) \cdot x = a \cdot x + b \cdot x$,
(iii) $(ab) \cdot x = a \cdot (b \cdot x)$,
(iv) $1 \cdot x = x$.

In fact, every abelian group corresponds to a module over $\mathbb{Z}$ (a $\mathbb{Z}$-module) by $n \cdot x \mapsto x + x + \cdots + x$.

Definition 1.1.6. We say that an $\mathscr{R}$-module is *finitely generated* if there exists finitely many elements $x_1, \ldots, x_n \in M$ such that for any $y \in M$ there exists $a_1, \ldots, a_n \in \mathscr{R}$ such that

$$y = a_1 \cdot x_1 + \cdots + a_n \cdot x_n.$$

Definition 1.1.7. Let M be an $\mathscr{R}$-module. If $N \subseteq M$ is a subgroup of M such that $\mathscr{R} \cdot N \subseteq N$, we say that N is a *submodule* of M.

Definition 1.1.8. Let M be a finitely generated module over an integral domain $\mathscr{I}$. The *rank* of M is the maximal number of $\mathscr{I}$-linearly independent elements in M. In other words, the rank of M is the maximal number n of elements $x_1, \ldots, x_n \in M$ such that if $y_1, \ldots, y_n \in \mathscr{I}$ and $y_1x_1 + \ldots + y_nx_n = 0$ then $y_i = 0$ for all $i = 1, \ldots, n$.

We introduce the concept of orders in number fields.

Definition 1.1.9. Let K be a number field. An *order* R in K is a subring of K which as a $\mathbb{Z}$-module is finitely generated and of maximal rank $\deg(K)$.

For example, $\mathbb{Z}[i] = \{a + bi : a, b \in \mathbb{Z}\}$ is an order in the number field $K = \mathbb{Q}(\sqrt{-1})$. It is clear that $\mathbb{Z}[i]$ is a subring of K. Furthermore, $\mathbb{Z}[i]$ considered as a $\mathbb{Z}$-module is generated by 1 and i and has rank $2 = \deg(K)$.

The set $\mathbb{Z}[i]$ is called the Gaussian integers and is in fact the ring of integers of $\mathbb{Q}(i)$. It is true in general that the ring of integers $\mathbb{Z}_K$ of any number field K is an order in K. Moreover, $\mathbb{Z}_K$ is the unique maximal order in K [4, p. 186].

Orders give us the definition of fractional ideals.

Definition 1.1.10. Let K be a number field and let R be an order in K. A *fractional ideal* $\mathcal{I}$ of R is a non-zero R-submodule of K such that there exists a non-zero integer d with $d\mathcal{I}$ being an ideal of R.

We are now ready to define the ideal class group.

Definition 1.1.11. Let K be a number field and $\mathbb{Z}_K$ its ring of integers. We say that two fractional ideals, $\mathcal{I}$ and $\mathcal{J}$, of $\mathbb{Z}_K$ are *equivalent* if $\exists \alpha \in K \setminus \{0\}$ such that $\alpha\mathcal{I} = \mathcal{J}$. This defines an equivalence relation $\sim$ on the set of all fractional ideals of $\mathbb{Z}_K$. The set of equivalence classes $\{[\mathcal{I}] : \mathcal{I}$ is a fractional ideal of $\mathbb{Z}_K\}$ is called the *ideal class group* of K, and is denoted by $Cl(K)$.

The set $Cl(K)$ is indeed a group under the operation $[\mathcal{I}] * [\mathcal{J}] = [\mathcal{I}\mathcal{J}]$, where (the fractional ideal of $\mathbb{Z}_K$) $\mathcal{I}\mathcal{J}$ is defined as

$$\mathcal{I}\mathcal{J} \coloneqq \left\{ \sum_{i=1}^{n} x_i y_i : x_i \in \mathcal{I}, y_i \in \mathcal{J}, n \in \mathbb{N} \right\}.$$

The group $Cl(K)$ is finite and commutative, and its order is referred to as the class number of K, see [4, p. 208]. For clarity, we repeat this below.

Theorem 1.1.1. Let K be a number field. The class group $Cl(K)$ is a finite abelian group.

Definition 1.1.12. Let K be a number field. The order of $Cl(K)$ is called the *class number* of K.

Recall the fundamental theorem for finite abelian groups: every finite abelian group is isomorphic to a direct product of cyclic groups of prime-power order.

We turn our focus to an important invariant connected to number fields, namely the discriminant. Moreover, we limit our scope to quadratic number fields.

Definition 1.1.13. Consider $d \in \mathbb{Z}$ such that $K = \mathbb{Q}(\sqrt{d})$ is a quadratic field. The *discriminant* $d(K)$ of K is defined to be

$$d(K) = \begin{cases} d \text{ if } d \equiv 1 \mod 4 \\ 4d \text{ otherwise .} \end{cases}$$

In the continuation of this paper, we solely consider so called fundamental discriminants.

Definition 1.1.14. If $d \in \mathbb{Z}$ is the discriminant of a quadratic field, then we say that d is a *fundamental discriminant*.

Thus, $d \in \mathbb{Z}$ is a fundamental discriminant if and only if

$$\begin{aligned} &(i) \ d \equiv 1 \mod 4 \text{ and } d \neq 1 \text{ is square-free, or} \\ &(ii) \ d = 4m \text{ where } m \equiv 2, 3 \mod 4 \text{ is square-free.} \end{aligned}$$

In case (i), $\mathbb{Q}(\sqrt{d})$ is the corresponding quadratic field and in (ii), the quadratic field is $\mathbb{Q}(\sqrt{d}) = \mathbb{Q}(\sqrt{m})$.

A different, yet (for imaginary quadratic fields) equivalent formulation of the class group, is given in Appendix A. This approach involves binary quadratic forms $f(x, y) = ax^2 + bxy + cy^2$ with discriminant $\Delta := b^2 - 4ac$; the class group in this setting is referred to as *the form class group*. Importantly and interestingly, if an integer $d < 0$ is square-free then the ideal class group of $K = \mathbb{Q}(\sqrt{d})$ is isomorphic to the form class group of discriminant $\Delta = d(K)$.

In this paper, we are only concerned about class groups of imaginary quadratic fields. If $d < 0$ is a fundamental discriminant, we shall denote, by $H(d)$, the ideal class group of $\mathbb{Q}(\sqrt{d})$. Moreover, we let $h(d) := |H(d)|$ represent the class number of $\mathbb{Q}(\sqrt{d})$.

Table 1.1 shows examples of related discriminants, quadratic fields, class numbers and class groups.

Table 1.1: Examples of fundamental discriminants $d < 0$, quadratic fields, class numbers and class groups. These examples were computed through PARI's `quadclassunit` [5]. Moreover, $\mathbb{Z}_n = (\mathbb{Z}_n, +) \cong C_n$.

d	Quadratic field	$h(d)$	$H(d)$
-3	$\mathbb{Q}(\sqrt{-3})$	1	$\mathbb{Z}_1$
-15	$\mathbb{Q}(\sqrt{-15})$	2	$\mathbb{Z}_2$
-24	$\mathbb{Q}(\sqrt{-6})$	2	$\mathbb{Z}_2$
-39	$\mathbb{Q}(\sqrt{-39})$	4	$\mathbb{Z}_4$
-555	$\mathbb{Q}(\sqrt{-555})$	4	$\mathbb{Z}_2 \times \mathbb{Z}_2$
-1031	$\mathbb{Q}(\sqrt{-1031})$	35	$\mathbb{Z}_{35}$
-2543	$\mathbb{Q}(\sqrt{-2543})$	35	$\mathbb{Z}_{35}$
-8932	$\mathbb{Q}(\sqrt{-2233})$	16	$\mathbb{Z}_4 \times \mathbb{Z}_2 \times \mathbb{Z}_2$
-12451	$\mathbb{Q}(\sqrt{-12451})$	25	$\mathbb{Z}_5 \times \mathbb{Z}_5$
-28771	$\mathbb{Q}(\sqrt{-28771})$	25	$\mathbb{Z}_{25}$
-29443	$\mathbb{Q}(\sqrt{-29443})$	25	$\mathbb{Z}_{25}$

1.2 Background

One of the most remarkable results regarding class numbers is Dirichlet's class number formula. Let $d < 0$ be a fundamental discriminant and set

$$w = \begin{cases} 2 \text{ if } d < -4, \\ 4 \text{ if } d = -4, \\ 6 \text{ if } d = -3. \end{cases}$$

In other words, w is the number of roots of unity in $\mathbb{Q}(\sqrt{d})$. The following theorem was proved by Dirichlet in 1839 [6].

Theorem 1.2.1.

$$h(d) = \frac{w\sqrt{|d|}}{2\pi} L(1, \chi_d).$$

In theorem 1.2.1, $L(1, \chi_d)$ is a Dirichlet L-series, which may be expressed

as

$$L(1,\chi_d) = \sum_{n=1}^{\infty} \frac{1}{n}\chi_d(n) = \prod_p \left(1 - \frac{1}{p}\chi_d(p)\right)^{-1},$$

where $\chi_d(\cdot)$ is the Kronecker symbol and the product is taken over prime numbers p. We remark that $L(1,\chi_d)$ is conditionally convergent [4, p. 237, 250]. For $a \in \mathbb{Z}$ we have $\chi_a(1) \coloneqq 1$ and

$$\chi_a(2) \coloneqq \begin{cases} 0 \text{ if } a \text{ is even,} \\ 1 \text{ if } a \equiv \pm 1 \mod 8, \\ -1 \text{ if } a \equiv \pm 3 \mod 8, \end{cases} \qquad \chi_a(-1) \coloneqq \begin{cases} -1 \text{ if } a < 0, \\ 1 \text{ if } a \geq 0. \end{cases}$$

The Kronecker symbol is a generalization of the Legendre symbol. Let $m \in \mathbb{Z}$ and write $m = up_1^{e_1} \cdots p_k^{e_k}$, where $u \in \{-1, 1\}$ and p_i is prime. Then, for an integer a,

$$\chi_a(m) = \chi_a(u) \prod_{i=1}^{k} \left(\frac{a}{p_i}\right)^{e_i},$$

where, for odd p_i, $(\frac{a}{p_i})$ is the Legendre symbol and $(\frac{a}{2}) = \chi_a(2)$.

To define the Legendre symbol, we need the notion of quadratic residues.

Definition 1.2.1. Let p be an odd prime and let $a \in \mathbb{Z}$. If there is an integer $n \in (0, p)$ such that

$$n^2 \equiv a \mod p,$$

we say that a is a *quadratic residue* modulo p. Otherwise, we say that a is a *quadratic nonresidue* modulo p.

The Legendre symbol $(\frac{a}{p})$ assumes three possible values.

$$\left(\frac{a}{p}\right) = \begin{cases} 1 \text{ if } a \text{ is a quadratic residue modulo } p \text{ and } a \not\equiv 0 \mod p, \\ -1 \text{ if } a \text{ is a quadratic nonresidue modulo } p, \\ 0 \text{ if } a \equiv 0 \mod p. \end{cases}$$

This means that $(\frac{23}{7}) = 1$ since $4^2 \equiv 23 \mod 7$.

Theorem 1.2.1 tells us the class number corresponding to the fundamental discriminant $d < 0$. A follow-up question is: how many fundamental

discriminants $d < 0$ maps to a given h? Consider a positive integer h and let

$$\mathcal{F}(h) = |\{\text{fundamental discriminants } d < 0 : h(d) = h\}|.$$

Table 1.1 shows that $\mathcal{F}(25) \geq 3$.

The quantity $\mathcal{F}(h)$ is closely related to the classical Gauss' class number problem for imaginary quadratic fields: given a positive integer h, determine all imaginary quadratic fields $\mathbb{Q}(\sqrt{d})$ with class number h.

The Gauss' class number problem was solved for all $h \leq 100$ by Watkins in 2003 [7]. However, the problem for $h = 1$ was solved already in 1952 by Heegner [8]. It is in fact known, in principle, how to solve the class number problem for any h due to Goldfeld and Gross-Zagier [8]. Nevertheless, it appears to be a very difficult problem from a computational point of view as h grows [7].

The negative fundamental discriminants of class number 1 are -3, -4, -7, -8, -11, -19, -43, -67 and -163, and thus we have $\mathcal{F}(1) = 9$. A few more examples, picked from Watkins tables, are $\mathcal{F}(25) = 95$, $\mathcal{F}(48) = 1365$, $\mathcal{F}(65) = 164$ and $\mathcal{F}(92) = 1248$.

In this paper, we restrict ourselves to only consider odd class numbers h, which is equivalent to only considering discriminants that are prime [9]. Figure 1.1 shows $\mathcal{F}(h)$ for all odd h in the range $1 - 100$.

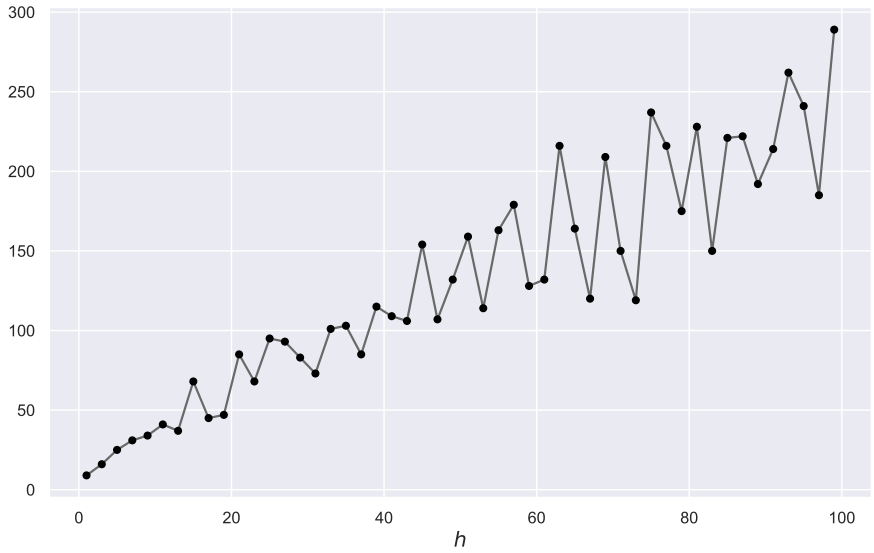


Figure 1.1: $\mathcal{F}(h)$ for all odd h in $[1, 100]$.

1.3 Research Questions

In this thesis we consider two main research questions. The first question concerns predicting $\mathcal{F}(h)$ based on h, while the other question is about predictions regarding $h(d)$ given d.

Question 1: Can deep learning models compete with state-of-the-art theory in predicting $\mathcal{F}(h)$ given h, and if so, under what circumstances?

Question 2: Given a fundamental prime discriminant $d < 0$, what can machine learning models tell us about the class number $h(d)$?

A detailed description of all underlying questions and problems is given in chapter 3.

1.4 Structure of the Thesis

Chapter 1 presents relevant background information about class groups and their class numbers, as well as the research questions of the thesis. In chapter 2 we present some related work in class group theory and machine learning. Chapter 3 presents the methods used to answer the research questions, while chapter 4 contains the experimental results. The thesis is concluded in chapters 5 and 6 with discussions, conclusions and ideas for future work.

Chapter 2

Related Work

This section provides a review of existing literature relevant to our research questions.

2.1 Class Group Heuristics

A significant part of all research concerning class groups of quadratic fields has resulted in conjectures and not in theorems. An important set of such heuristically derived conjectures is the Cohen-Lenstra heuristics (see [4, p. 295-297] for a detailed introduction). For instance, if p is an odd prime, the Cohen-Lenstra heuristics states that p divides a random class number $h(d)$ ($d < 0$) with probability $1 - (p)_\infty$, where

$$(p)_\infty := \prod_{k=1}^{\infty} \left(1 - \frac{1}{p^k}\right).$$

This is an interesting observation. At first one might think that 3 divides a random $h(d)$ with probability $1/3$, but the Cohen-Lenstra heuristics suggests that the actual probability is approximately 0.43987.

Interestingly, the Cohen-Lenstra heuristics tell us that if $d < 0$ is a fundamental prime discriminant, then $H(d)$ is cyclic with very high probability (see [4, p. 261]). Thus, looking back at table 1.1, discriminants d with $h(d) = 25$ should much more often have $H(d) = \mathbb{Z}_{25}$ rather than $H(d) = \mathbb{Z}_5 \times \mathbb{Z}_5$. *

Regarding the quantity $\mathcal{F}(h)$, K. Soundararajan conjectured in 2007 [10] that $\mathcal{F}(h) \asymp h/\log(h)$ for odd h. This means that $\mathcal{F}(h) = \mathcal{O}(h/\log(h))$ and

*Recall that $C_n \times C_m$ is cyclic if and only if n and m are coprime.

$h/\log(h) = \mathcal{O}(\mathcal{F}(h))$.

In their work [9] from 2014, Holmin, Jones, Kurlberg, McLeman and Petersen refined Soundararajan's conjecture regarding the asymptotic behaviour of $\mathcal{F}(h)$. Let

$$\mathbb{Y}(p) := \begin{cases} 1 \text{ with probability } 1/2 \\ -1 \text{ with probability } 1/2, \end{cases}$$

and consider the random Euler product $L(1,\mathbb{Y}) := \prod_p \left(1 - \frac{\mathbb{Y}(p)}{p}\right)^{-1}$. The authors conjectured that

$$\mathcal{F}(h) \sim \pi^2 \cdot \frac{\mathfrak{C}}{15} \cdot \mathfrak{c}(h) \cdot h \cdot \mathbb{E}\left(\frac{1}{L(1,\mathbb{Y})^2 \log(\pi h / L(1,\mathbb{Y}))}\right) \tag{2.1}$$

as $h \to \infty$ through odd values.

In conjecture 2.1, the values $\mathfrak{C}$ and $\mathfrak{c}(h)$ are defined as

$$\mathfrak{C} := 15 \prod_{\substack{\ell=3 \\ \ell \text{ prime}}}^{\infty} \prod_{i=2}^{\infty} \left(1 - \frac{1}{\ell^i}\right) \approx 11.317,$$

and

$$\mathfrak{c}(h) := \prod_{p^n || h} \prod_{i=1}^{n} \left(1 - \frac{1}{p^i}\right)^{-1}.$$

Thus, the quantity $\mathfrak{c}(h)$ encapsulates divisibility properties of h. For instance, consider two odd class numbers h_1 and h_2 of relatively large size, say around 10^6, where h_1 contains many small prime factors — suppose for example that $3^4 \mid h$ — and h_2 is itself a prime number. In this case it is easy to reach the conclusion that h_1 is far more 'divisible' than h_2 by merely comparing $\mathfrak{c}(h_1)$ with $\mathfrak{c}(h_2)$. Since $\mathfrak{c}(h) = \prod_{p^n||h} \prod_{i=1}^{n} (p^i/p^i - 1)$ we have that $\mathfrak{c}(h_2)$ is close to one wheras $\mathfrak{c}(h_1) > 1.77$. A larger $\mathfrak{c}(h)$-value implies divisibility by more small prime numbers.

The authors of [9] used 2.1 to construct a formula $\mathrm{pred}(h)$ for approximating $\mathcal{F}(h)$ for odd h and they visualized the performance of $\mathrm{pred}(h)$ by plotting a histogram of the scaled errors $r(h) := (\mathcal{F}(h) - \mathrm{pred}(h))/\sqrt{\mathrm{pred}(h)}$ (see figure 3.1 for $k = 3$). The interesting thing about this error distribution is that there are two distinct peaks where the left one consists mostly of inputs h that are divisible by three whereas the right peak

corresponds to inputs not divisible by three (see [9] for these results). This intriguing three divisibility bias is something we keep in mind when designing experiments and analyzing results.

2.2 Machine Learning in Number Theory

This overview is divided into two parts: one that covers the important aspect of feature engineering in machine learning and one that discusses the general paradigm of computers and AI within math and number theory.

Feature engineering

In their paper [11] from 2018, Stekel, Shukrun and Azaria used machine learning to predict Goldbach's partition function. Goldbach's partition function is related to Goldbach's conjecture, which is one of the most famous unsolved problems in number theory. The conjecture was proposed in 1742 by Christian Goldbach and states that every even integer greater than 2 can be written as the sum of two prime numbers. To this day, no one has been able to prove or disprove this simple statement. Nevertheless, the authors of the 2018 paper tried to approximate Goldbach's partition function

$$G(n) := \sum_{\substack{p,q \text{ primes} \\ p \leq q}} \mathbb{1}_{\{p+q=n\}},$$

which essentially counts the number of prime pairs that sum to a given integer n, through the use of deep learning models. They found that by converting the input integers to their binary, ternary (base 3), quinary (base 5) and septenary (base 7) forms — thus representing an integer as a high-dimensional feature vector — the models were able to outperform all previous, analytically derived, prediction formulas.

The Goldbach paper showed that deep learning models can achieve great performance in approximating $G(n)$. However, this hinges on the extraction of relevant features from the natural number n. The same is true for all problems in machine learning — without relevant features, the models do not perform. This begs the question: which features are fruitful as input to a model? The answer to this question is not always obvious and of course depends on the nature of the problem.

The process of producing a feature space for some data variable is referred to as feature engineering. For instance, consider a streaming service

provider aiming to predict subscription cancellations. Initially, representing customers solely by their customer-IDs yields a one-dimensional feature space, likely insufficient for robust predictions. Instead, a more sophisticated approach involves extracting relevant features from customer data, such as age, gender, location, subscription duration and type, billing history, and platform engagement metrics like ratings given. This enriches the dataset, creating a more complex, high-dimensional customer feature space. Neural networks excel in leveraging such intricate data representations, making them well-suited for tasks requiring analysis of multifaceted relationships and patterns within the data.

An article that analyzes the impact of feature engineering in deep learning, especially when it comes to problems in number theory, is [12] written by Wu, Yang, Ahsan and Wang. They consider the task of classifying integers based on their residues when divided by small prime numbers. For instance, when dividing integers by 2 the problem becomes a binary classification task: input=n, label=$n \mod 2$. They analyze the performance of different deep learning models over varying feature spaces for the integers n. They conclude that the provided feature space has great impact on the results. For example, consider the task of predicting $n \mod 3$. In this case we expect models with enough degrees of freedom to reach 1.0 accuracy when given the ternary form of n because we thereby essentially provide the models with the correct answer already in the feature vector. On the other hand, providing the models with n in its binary form yields an expected accuracy of ~ 0.33.

Another paper that demonstrates the importance of feature engineering in machine learning is [13] by Heaton. They show how various machine learning models respond differently to the same feature space. Moreover, they stress that if a model can learn a feature on its own, there is no need to include it in the feature vector. These studies highlight the fact that manual feature engineering can be both time-consuming and non-intuitive.

AI in mathematics

In his work [14] from 2021, Yang-Hui He reviews and summarizes recent experiments aimed at machine learning mathematical data. He brings up scenarios where machine learning models evidently do well, for example in classifying graphs as acyclic or not, or as planar or not based on the corresponding adjacency matrix as input. Moreover, he highlights the expected difficulty in problems belonging to number theory; we do not expect machine learning algorithms to easily detect basic new patterns in the primes.

An interesting aspect of this ML-paradigm of mathematics, as discussed in [14], is the potential aid in producing conjectures. Suppose a neural network surprisingly achieves 1.0 accuracy in a classification task related to mathematical data. This can certainly be useful in itself since we are now in the possession of a model that can tell us which class an object belongs to. More interesingly though is if the model is interpretable or not. If it is possible to extract some association rule constructed by the network, then we have a potential conjecture. However, powerful ML models such as deep neural networks are notorious for being difficult to interpret due to their black-box nature. In his book [15] from 2021, Cristoph Molnar discusses the importance of interpretable machine learning. Oftentimes we need to know why a model produces a certain output.

The abovementioned way of doing mathematics — arriving at conjectures and possible theorems from experiments and observations — is referred to, in [14], as a top-down approach. The other way, namely starting from a theoretical foundation and directly build on it, is referred to as a bottom-up approach. Intuitively, the latter approach might feel as the natural way of doing mathematics. Altough, countless examples show that oftentimes knowledge gained from experiments and observations precedes formalization (see [14] for interesting notes).

In regards to bottom-up math, computers have played an increasingly integral role in research endeavors over the past few decades. Formal proof management systems such as Coq [16] and Lean [17] have enabled mathematicians to explore complex mathematics in a different way. The use of computers to build and verify mathematics through formalized logical principles will likely keep increasing. To name one of many already accomplished feats, in 2005, Georges Gonthier and Benjamin Werner were able to use Coq to formalize a proof of the famous 4-color theorem [18].

More recently, AI systems that intertwine formal logic environments with generative transformers have demonstrated promising capabilities in generating mathematical proofs [19].

In summary, computers and AI show potential in aiding mathematicians in regards to both approaches mentioned avove.

Chapter 3

Methodology

As mentioned in section 2.2, it can be tricky to answer in advance the question whether a machine learning model 'should' be able to solve a problem related to mathematical data based on some given feature space. Sometimes — due to human knowledge in the area — the answer is obvious, but other times the question can be really daunting. In many cases, it simply boils down to a lot of trial and error.

When designing the experiments purposed to give insight into questions 1 and 2, we make efforts to interpret the models, when they perform well. We do this by thoughtful feature engineering, examining feature importance and comparing the results with appropriate benchmarks.

For quantifying feature importance, we use the classical permutation feature importance [15]. Suppose we have trained and evaluated a model $\mathscr{M}$ on a task and have gotten the score $\mathscr{S}$. If $\mathscr{M}$ is a classifier (a model used to separate data points into categorical classes), the score $\mathscr{S}$ would be accuracy; if $\mathscr{M}$ is a regression model (a model used to approximate a continuous target variable), $\mathscr{S}$ would be the R^2. Recall that if y_i denote labels with mean value $\overline{y}$ and f_i denote the approximations of y_i, then the R^2-score is defined as

$$R^2 := 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \overline{y})^2}.$$

We want to measure the contribution of each feature to the model's performance. Let $M \in \mathbb{R}^{n \times m}$ represent our evaluation data set; the m columns represent the features and in total we have n data points. To compute the importance score s_j of feature $j \in \{1, \ldots, m\}$, generate the matrix M^{perm_j} by permuting column j of M. This permutation should randomize the values of the feature j while preserving the same distribution of values within the

column, thus this process should break the link between feature j and the labels. This will give a new evaluation score $\mathscr{S}_j$. We can repeat this process k times so that we get permuted scores $\mathscr{S}_{j,1}, \ldots, \mathscr{S}_{j,k}$. We can then define the importance score of feature j as

$$s_j = \mathscr{S} - \frac{1}{k}\sum_{i=1}^{k} \mathscr{S}_{j,k}.$$

The objective of this research is not to develop efficient ML algorithms for practical computations of class numbers or $\mathcal{F}(h)$. Instead, the primary aim is to explore the question: do discernible patterns exist within the data? Consequently, we refrain from imposing explicit constraints on the construction of feature spaces for the input data. Our intention is to explore the response of various models to these distinct feature spaces.

3.1 Predicting $\mathcal{F}(h)$

This section presents the methods used to answer question 1. In particular, our main task is to implement supervised deep learning methods for approximating $\mathcal{F}(h)$ given h.

3.1.1 Data Collection

We collect data points of the form $(h, \mathcal{F}(h))$ for all odd natural numbers $h < 10^6$ from [20]. *

3.1.2 Problem Setup and Model Design

We consider the problem of predicting $\mathcal{F}(h)$, given h, as a regression task. Let $\mathcal{T}^h : \mathbb{Z}_+ \to [0,1]^\ell$ represent a function that maps each h to a feature vector with ℓ elements. We define the feature space corresponding to $\mathcal{T}^h$ as $\mathcal{D}_{\mathcal{T}^h} := \mathrm{im}(\mathcal{T}^h)$. Let $\mathscr{M}_\theta : \mathcal{D}_{\mathcal{T}^h} \to \mathbb{R}$ be our prediction model, with parameters $\theta \in \mathbb{R}^m$. If S is our input dataset, we define the corresponding loss function $\mathcal{L}_{S,\mathcal{T}^h} : \mathbb{R}^m \to \mathbb{R}$ as

$$\mathcal{L}_{S,\mathcal{T}^h}(\theta) = \frac{1}{|S|}\sum_{h_i \in S} |(\mathscr{M}_\theta \circ \mathcal{T}^h)(h_i) - \mathcal{F}(h_i)|^2 \quad \text{(MSE-loss)}$$

*The computations of $\mathcal{F}(h)$ assume the generalized Riemann hypothesis (GRH).

and the objective is to minimize $\mathcal{L}_{S,\mathcal{T}^h}$ with respect to θ.

The models $\mathscr{M}_\theta$ we consider for this task are deep neural networks, and in particular we employ multilayer perceptrons (MLPs). An MLP is a type of feedforward artificial neural network (ANN) that consists of at least three layers of neurons. An important component of an MLP is its *activation function*. The activation function is a function which is usually non-linear. A common choice of activation function is ReLU, which is defined by $\text{ReLU}(x) = \max(0, x)$. An arbitrary MLP of n layers and activation function a can be expressed as

$$\mathscr{M}_\theta(x) = (L_n \circ a \circ L_{n-1} \circ a \cdots \circ a \circ L_1)(x), \tag{3.1}$$

where $L_i(w) = A_i w + b_i$; the A_i are linear transformations (matrices) and the b_i are bias vectors. The θ-vector represents the learnable parameters of the model, i.e. the elements of all A_i and b_i.

To investigate how the network architecture affects performance, we consider two different models, call them $\mathscr{M}_{\theta_1}$ and $\mathscr{M}_{\theta_2}$. The explicit architecure of $\mathscr{M}_{\theta_1}$ and $\mathscr{M}_{\theta_2}$ are given in table 3.1 and 3.2, respectively. For both models, we use ReLU as activation function.

Table 3.1: Architecture of model $\mathscr{M}_{\theta_1}$. This model has 5 hidden layers and 60721 learnable parameters.

Layer	**Output Size**	**# Parameters**	**Activation**
Input	20		
Linear	120	2520	ReLU
Linear	120	14520	ReLU
Linear	120	14520	ReLU
Linear	120	14520	ReLU
Linear	120	14520	ReLU
Output	1	121	

It is of interest to examine how the performances of the neural network models change as we provide them with different representations of the inputs h. For example, will a model perform significantly better when given the values $\mathfrak{c}(h)$ attached to each h? Does a model's performance depend on whether it knows if h is divisible by 3 or not? To try to answer these questions, we consider three feature spaces: $\mathcal{D}_{\mathcal{T}_A^h}$, $\mathcal{D}_{\mathcal{T}_B^h}$ and $\mathcal{D}_{\mathcal{T}_C^h}$, specified below.

Table 3.2: Architecture of model $\mathscr{M}_{\theta_2}$. This model has 11 hidden layers and 56571 learnable parameters.

Layer	Output Size	# Parameters	Activation
Input	20		
Linear	160	3360	ReLU
Linear	160	25760	ReLU
Linear	80	12880	ReLU
Linear	80	6480	ReLU
Linear	40	3240	ReLU
Linear	40	1640	ReLU
Linear	40	1640	ReLU
Linear	20	820	ReLU
Linear	20	420	ReLU
Linear	10	210	ReLU
Linear	10	110	ReLU
Output	1	11	

Some notations we use are

$$\begin{aligned}
\Omega(h) &= \sum_{p^n||h} n \text{ (the number of prime factors of } h\text{)},\\
\omega(h) &= \sum_{p|h} 1 \text{ (the number of distinct prime factors of } h\text{)},\\
h &= \sum_{i\in\mathbb{N}} c_{\mathfrak{B},i}(h)\mathfrak{B}^i \text{ (base-}\mathfrak{B}\text{ representation of } h)\\
&\quad c_{\mathfrak{B},i}(h) \in \{0,1,\ldots,\mathfrak{B}-1\},\ \mathfrak{B}\in\mathbb{Z}_{\geq 2}.
\end{aligned}$$

Below when we write $c_{\mathfrak{B},a-b}(h)$ with $0 < a < b$ we mean the features $c_{\mathfrak{B},a}(h), c_{\mathfrak{B},a+1}(h), \ldots, c_{\mathfrak{B},b}(h)$.

The feature vectors corresponding to each feature space will have a constant size of 20 elements, i.e. $\mathcal{D}_{\mathcal{T}^h_{A,B,C}} \subseteq [0,1]^{20}$. The common features for all spaces are h, $\log(\pi h)$, $\Omega(h)$, $\omega(h)$, $c_{2,1-5}(h)$, $c_{5,0-2}(h)$ and $c_{7,0-1}(h)$. *

In addition to the common features, $\mathcal{D}_{\mathcal{T}^h_A}$ contains $c_{2,6-7}(h)$, $c_{5,3-4}(h)$ and $c_{7,2-3}(h)$. Thus, $\mathcal{D}_{\mathcal{T}^h_A}$ contains neither $\mathfrak{c}(h)$ nor $h \mod 3$. In $\mathcal{D}_{\mathcal{T}^h_B}$, we include $\mathfrak{c}(h)$ as well as $c_{2,6}(h)$, $c_{5,3-4}(h)$ and $c_{7,2-3}(h)$. Lastly, in $\mathcal{D}_{\mathcal{T}^h_C}$ we have $\mathfrak{c}(h)$ and $c_{3,0-4}(h)$.

*All features are normalized.

3.1.3 Benchmarks

When predicting $\mathcal{F}(h)$, we benchmark our models against conjecture 2.1. To do this, we evaluate the expected value $\mathbb{E}\left[1/(L(1,\mathbb{Y})^2 \log(\pi h/L(1,\mathbb{Y})))\right]$. To begin with, we have

$$\log\left(\frac{\pi h}{L(1,\mathbb{Y})}\right) = \log(\pi h) - \log(L(1,\mathbb{Y})) = \log(\pi h)\left(1 - \frac{\log(L(1,\mathbb{Y}))}{\log(\pi h)}\right).$$

Moreover, by using a Taylor expansion we may write

$$\frac{1}{1-\frac{\log(L(1,\mathbb{Y}))}{\log(\pi h)}} = \sum_{k=0}^{\infty}\left(\frac{\log(L(1,\mathbb{Y}))}{\log(\pi h)}\right)^k.$$

Thus we obtain

$$\begin{aligned}\mathbb{E}\left(\frac{1}{L(1,\mathbb{Y})^2\log(\pi h/L(1,\mathbb{Y}))}\right) &= \frac{1}{\log(\pi h)}\mathbb{E}\left(\frac{1}{L(1,\mathbb{Y})^2}\sum_{k=0}^{\infty}\left(\frac{\log(L(1,\mathbb{Y}))}{\log(\pi h)}\right)^k\right)\\ &= \frac{1}{\log(\pi h)}\sum_{k=0}^{\infty}\frac{1}{\log(\pi h)^k}\mathbb{E}\left(\frac{\log(L(1,\mathbb{Y}))^k}{L(1,\mathbb{Y})^2}\right).\end{aligned}$$

By denoting

$$\begin{aligned}E_k &\coloneqq \mathbb{E}\left(\frac{\log(L(1,\mathbb{Y}))^k}{L(1,\mathbb{Y})^2}\right), \text{ and}\\ c_k &\coloneqq E_k/E_0,\end{aligned}$$

we get

$$\begin{aligned}\mathcal{F}(h) &\sim \pi^2\cdot\frac{\mathfrak{C}}{15}\cdot\mathfrak{c}(h)\cdot\frac{h}{\log(\pi h)}\left(E_0+\frac{E_1}{\log(\pi h)}+\frac{E_2}{\log(\pi h)^2}+\frac{E_3}{\log(\pi h)^3}+\dots\right)\\ &= E_0\cdot\pi^2\cdot\frac{\mathfrak{C}}{15}\cdot\mathfrak{c}(h)\cdot\frac{h}{\log(\pi h)}\left(1+\frac{c_1}{\log(\pi h)}+\frac{c_2}{\log(\pi h)^2}+\dots\right).\end{aligned}$$

Let $L_p \coloneqq 1-\mathbb{Y}(p)/p$. We have $E_0 = \mathbb{E}\left(\prod_p L_p^2\right) = \prod_p \mathbb{E}\left(L_p^2\right) = 15/\pi^2$, and for our benchmarks, we explicitly compute c_1, c_2, c_3 and c_4 (see Appendix

B). To summarise, we have

$$\begin{aligned} c_1 &\approx -0.578072, \\ c_2 &\approx +0.604050, \\ c_3 &\approx -0.526259, \\ c_4 &\approx +0.618741. \end{aligned}$$

We define the predictive formula, pred_k, corresponding to including k coefficients of the expansion as

$$\mathrm{pred}_k(h) := \mathfrak{C} \cdot \mathfrak{c}(h) \cdot \frac{h}{\log(\pi h)} \sum_{j=0}^{k} \frac{c_j}{\log(\pi h)^j}.$$

These formulas provide natural benchmarks for our experiments. To visualize the performance of the benchmark formulas, define for each k the scaled error function $r_k(h) := (\mathcal{F}(h) - \mathrm{pred}_k(h))/\sqrt{\mathrm{pred}_k(h)}$, as well as the sample

$$\mathcal{S}_k := \{r_k(h) : h \in [5 \cdot 10^5, 10^6] \text{ is odd}\}.$$

In figure 3.1 we present histograms of $\mathcal{S}_k$ for $k = 0, 1, 2, 3, 4$. For each histogram, we include the sample mean μ and standard deviation σ. Moreover, for each k we include, in the legend, the corresponding mean squared error (MSE) $\frac{1}{2.5 \cdot 10^5} \sum_{h \in [5 \cdot 10^5, 10^6]} |\mathcal{F}(h) - \mathrm{pred}_k(h)|^2$, where the sum is taken over odd integers h.

3.1.4 Experiments

We train both models, $\mathscr{M}_{\theta_1}$ and $\mathscr{M}_{\theta_2}$, on each feature space, $\mathcal{D}_{\mathcal{T}_A^h}$, $\mathcal{D}_{\mathcal{T}_B^h}$ and $\mathcal{D}_{\mathcal{T}_C^h}$, separately. From this we get six different trained models. For each model and each feature space, we train for 2500 epochs (an epoch corresponds to one complete pass of the training dataset through the model) and we employ the ADAM optimizer [21] with hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate $\mathtt{lr}$ is initially set to $5e-03$ and decays exponentially with factor $\gamma = 0.90$ after every epoch. When $\mathtt{lr}$ has reached a value below $5e-05$, the decay halts and training proceeds with constant learning rate. In other words, with $\mathtt{lr}(0) = 5e-03$ we have, for epochs $t = 1, \ldots, 2500$,

$$\mathtt{lr}(t) = \begin{cases} \mathtt{lr}(t-1) \cdot \gamma \text{ if } \mathtt{lr}(t-1) > 5 \cdot 10^{-5}, \\ \mathtt{lr}(t-1) \text{ otherwise.} \end{cases}$$

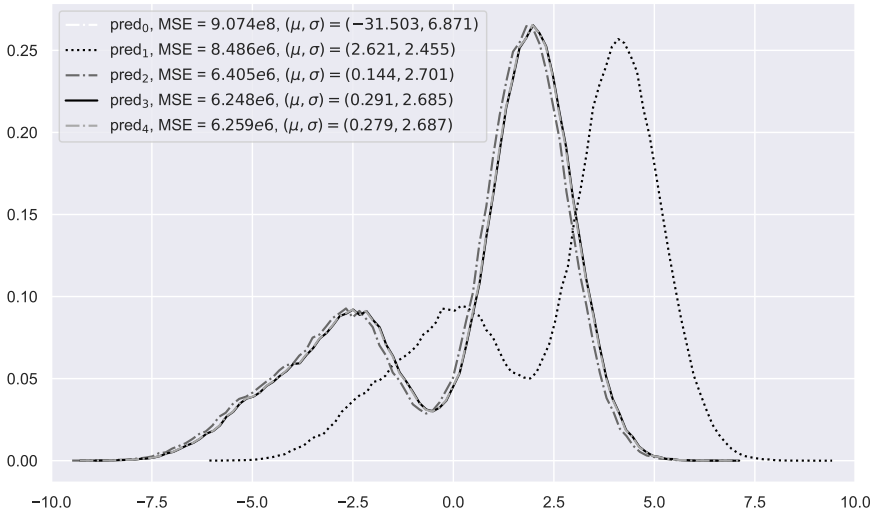


Figure 3.1: For $k = 0, 1, 2, 3, 4$, we plot a histogram of the error data $\mathcal{S}_k$. The μ and σ represent the corresponding sample mean and standard deviation.

For evaluation of the models, we consider the mean squared error (MSE), as well as the mean, μ, and standard deviation, σ, of the scaled errors $r_{\text{model}}(h) \coloneqq (\mathcal{F}(h) - \text{model}(h))/\sqrt{\text{model}(h)}$. This gives us three evaluation metrics which indicate the performance of the models.

In regards to training and test split, we consider two distinct experiments purposed to investigate interpolation vs extrapolation. In both experiments, the size of the training and test datasets are $4\text{e}5$ and $1\text{e}5$ respectively. For training, we use a batch size of 1000, meaning that the model parameters are updated after each processed batch of 1000 data points. Consequently, an epoch in this scenario corresponds to 400 gradient steps.

Interpolation

Let $\mathcal{B}$ denote our total input dataset, i.e. $\mathcal{B} = \{h \in [1, 10^6] : h \in \mathbb{Z} \wedge h \equiv 1 \mod 2\}$. We use K-fold cross-validation for evaluation, i.e. we shuffle $\mathcal{B}$ and divide it into K subsets, $\mathcal{B}_k$ $(k = 1, \ldots, K)$, such that

$$\left(|\mathcal{B}_k| = \frac{|\mathcal{B}|}{K} \forall k\right) \wedge (k \neq l \implies \mathcal{B}_k \cap \mathcal{B}_l = \emptyset) \wedge \left(\bigcup_{k=1}^{K} \mathcal{B}_k = \mathcal{B}\right).$$

Let $\mathcal{T}^h$ be the current feature function. For $k = 1, \ldots, K$, let $\mathscr{M}_{\theta(k)}$ denote the version of the model $\mathscr{M}_\theta$ that was trained on $\mathcal{B} \setminus \mathcal{B}_k$ and is going to be

evaluated on $\mathcal{B}_k$. For evaluation, we quantify the cross-validated performance as

$$\frac{1}{K}\sum_{k=1}^{K}\mathcal{L}_{\mathcal{B}_k,\mathcal{T}^h}\left(\theta(k)\right).$$

When conducting the experiments we set $K = 5$. The choice of K is arbitrary, however $K = 5$ implies that we iteratively train on 80% of the data and test on the other 20% which is a reasonable distribution in this context.

Extrapolation

The aim of this experiment is to investigate how well the models can extrapolate to inputs of greater size. Therefore, we train on all odd integers $h \in [1, 8 \cdot 10^5]$ and validate on all odd $h \in [8 \cdot 10^5, 10^6]$. Here we do not use cross-validation.

3.2 Predicting Class Numbers

This section presents the methods used to answer question 2. We limit our focus to making predictions regarding divisibility properties of class numbers $h(d)$ based on discriminant information. For this aim, we implement supervised machine learning methods for binary classification.

3.2.1 Data Collection

We use the Python package `CyPari2` to compute $h(d)$ for all fundamental prime discriminants $d < 0$ such that $|d| \in [10^6, 65 \cdot 10^6]$. The package `CyPari2` provides a Python interface for the computer algebra software PARI [5]. These computations give us a total of 1881326 data points of the form $(d, h(d))$. This collection acts as a pool from which we select data points that are appropriate in a particular context.

3.2.2 Problem Setup and Model Design

Let $\mathfrak{P}$ (positive samples) and $\mathfrak{N}$ (negative samples) represent two arbitrary distinct classes for fundamental prime discriminants $d < 0$. We consider the task of separating fundamental prime discriminants $d < 0$ based on divisibility properties in their corresponding class numbers $h(d)$. For instance, we investigate the ability of deep learning models to classify a random

discriminant d as having class number $h(d)$ such that either $3 \mid h$ or $3 \nmid h(d)$. In this example we would set

$$\mathfrak{P} \coloneqq \{\text{fundamental prime discriminants } d < 0 : 3 \mid h(d)\},$$
$$\mathfrak{N} \coloneqq \{\text{fundamental prime discriminants } d < 0 : 3 \nmid h(d)\}.$$

We stick to the notation from 3.1 and let $\mathcal{T}^d : \mathbb{Z}_- \to [0,1]^\ell$ represent a generic feature function that maps each d to a feature vector with ℓ elements. With parameters $\theta \in \mathbb{R}^n$, let $\mathscr{M}_\theta : \mathcal{D}_{\mathcal{T}^d} \to (0,1)$ denote a model that outputs the predicted probability that a given discrimant d belongs to class $\mathfrak{P}$. To each d, we attach a label $y_d = \mathbb{1}_{\{d \in \mathfrak{P}\}}$ (ground truth probability) and the loss function we want to minimize over our input dataset S is a binary cross-entropy loss $\mathcal{L}_{S,\mathcal{T}^d} : \mathbb{R}^n \to \mathbb{R}$

$$\mathcal{L}_{S,\mathcal{T}^d}(\theta) = -\frac{1}{|S|} \sum_{d_i \in S} [y_d \log((\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i)) + (1 - y_d) \log(1 - (\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i))].$$

For these experiments, we consider one neural network architecure, which we denote as $\mathscr{M}_\theta$ from here on out. In addition to this, we also review the performance of linear classifiers (see section 3.2.3). The model $\mathscr{M}_\theta$ is an MLP with input size $\ell = 54$ and four hidden layers of size 100. The output layer consists of one neuron. Hence, the model consists of 35901 learnable parameters. The activation function is ReLU for the hidden layers and sigmoid for the output layer. Recall the sigmoid function $\sigma : \mathbb{R} \to (0,1)$, defined by

$$\sigma(x) = \frac{e^x}{1 + e^x}.$$

Let d_i be a discriminant we want to classify as beloning to either $\mathfrak{P}$ or $\mathfrak{N}$ and recall that its label is $\mathbb{1}_{\{d_i \in \mathfrak{P}\}}$. Having σ as activation for the output neuron means that if

$$z(d_i) \coloneqq (\mathscr{M}_\theta^{\text{pre}} \circ \mathcal{T}^d)(d_i) \in \mathbb{R}$$

denotes the *pre-activation* value of the output neuron of the network (compare $\mathscr{M}_\theta^{\text{pre}}$ with 3.1), we pass this value through the sigmoid function so that the output of the model is $(\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i) = \sigma(z(d_i)) \in (0,1)$. Thus the sigmoid activation is particularly useful in the binary classification-scenario since it allows us to interpret the output of $(\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i)$ as the probability that d_i belongs to $\mathfrak{P}$. When computing the accuracy of the model, we convert these probabilities into binary predictions by $\text{pred}(d_i) \coloneqq \mathbb{1}_{\{(\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i) \geq 0.5\}}$. This is discussed in more detail below.

To see how the models (both the neural network and the linear models) perform for different discriminant representations, we try four different feature spaces, denoted $\mathcal{D}_{\mathcal{T}_A^d}$, $\mathcal{D}_{\mathcal{T}_B^d}$, $\mathcal{D}_{\mathcal{T}_C^d}$ and $\mathcal{D}_{\mathcal{T}_D^d}$.

We let $\mathcal{D}_{\mathcal{T}_A^d}$ be the most basic feature space — which will only be used for the linear model as baseline — consisting only of normalized discriminants $|d|$. Moreover, we let $\mathcal{D}_{\mathcal{T}_B^d}$ consist of $c_{2,2-25}(|d|)$, $c_{3,0-16}(|d|)$, $c_{5,0-11}(|d|)$ and $c_{7,0}(|d|)$. For the space $\mathcal{D}_{\mathcal{T}_C^d}$, we take inspiration from the class number formula 1.2.1 and let the ith feature be $\chi_d(p_i)$ where p_i is the ith prime for $i = 1, \dots, 54$. We make an important remark here that the symbol $\chi_d(\cdot)$ has the following property:

$$\chi_d(nm) = \chi_d(n)\chi_d(m).$$

Thus, if for instance the feature $\chi_d(35)$ is useful to the model, it should be able to easily construct it on its own by $\chi_d(35) = \chi_d(5) \cdot \chi_d(7)$.

For the space $\mathcal{D}_{\mathcal{T}_D^d}$, let $\pi(\cdot)$ denote the prime counting function. Moreover, we introduce the following notation.

$$\pi_d(|d|) \coloneqq \sum_{i=1}^{|d|} \mathbb{1}_{\{-i \text{ is a fundamental prime discriminant}\}}.$$

The space $\mathcal{D}_{\mathcal{T}_D^d}$ consists of $c_{2,2-5}(|d|)$, $c_{3,0-4}(|d|)$, $c_{5,0-4}(|d|)$, $c_{7,0-4}(|d|)$, $c_{10,0-4}(|d|)$, $c_{10,0-4}(10^4|d|^{\frac{1}{2}})$, $c_{2,0-1}(\pi(|d|))$, $c_{3,0-1}(\pi(|d|))$, $c_{5,0-1}(\pi(|d|))$, $c_{7,0-1}(\pi(|d|))$, $c_{10,0-1}(\pi(|d|))$, $c_{2,0-1}(\pi_d(|d|))$, $c_{3,0-1}(\pi_d(|d|))$, $c_{5,0-1}(\pi_d(|d|))$, $c_{7,0-1}(\pi_d(|d|))$ and $c_{10,0-1}(\pi_d(|d|))$, as well as the five features $\mathrm{chen}(|d|)$, $\mathrm{balanced}(|d|)$, $\mathrm{safe}(|d|)$, $\mathrm{sophie}(|d|)$, $\mathrm{super}(|d|)$ which indicate whether or not $|d|$ is a Chen prime or a balanced prime etc. The inclusion of these last five features in the feature space is motivated solely by an exploratory approach, driven by a curiosity to investigate whether these attributes might have any correlation with the labels.

3.2.3 Benchmarks

To the best of the author's knowledge, no similar experiments have been conducted. Therefore, we need to construct baseline performance benchmarks to establish a foundation for comparison. We do this by employing Fisher's linear discriminant analysis (LDA) (we follow the exposition in [22, p. 86]).

Fisher's LDA is a classical machine learning method that can be viewed as a linear classification model. Hence, this allows us to benchmark the neural

networks against linear classifiers.

Let $\mathcal{N} = \{\hat{d}_i\}$ be a collection of discriminant feature vectors, where $\hat{d}_i \in \mathcal{D}_{\mathcal{T}^d}$ for some feature space $\mathcal{D}_{\mathcal{T}^d}$. This $\mathcal{N}$ is our training set. Furthermore, let $\mathfrak{P}$ and $\mathfrak{N}$ represent two distinct classes such that every $\hat{d}_i$ belongs to either class $\mathfrak{P}$ or class $\mathfrak{N}$. The goal is to reduce the data into one dimension in a way such that the two classes $\mathfrak{P}$ and $\mathfrak{N}$ can be distinguished from each other.

Let

$$m_{\mathfrak{P}} \coloneqq \frac{1}{|\mathfrak{P}|} \sum_{d \in \mathfrak{P}} \hat{d}_i \text{ and } m_{\mathfrak{N}} \coloneqq \frac{1}{|\mathfrak{N}|} \sum_{d \in \mathfrak{N}} \hat{d}_i$$

denote the mean vectors corresponding to population $\mathfrak{P}$ and $\mathfrak{N}$ respectively. Moreover, let $\Sigma_{\mathfrak{P}}$ and $\Sigma_{\mathfrak{N}}$ be the corresponding covariance matrices

$$(\Sigma_{\mathfrak{P}})_{ij} = \text{cov}(\text{feat.}_i, \text{feat.}_j) \text{ and } (\Sigma_{\mathfrak{N}})_{ij} = \text{cov}(\text{feat.}_i, \text{feat.}_j).$$

Consider the separation ratio

$$R(x) = \frac{(x^T(m_{\mathfrak{P}} - m_{\mathfrak{N}}))^2}{x^T(\Sigma_{\mathfrak{P}} + \Sigma_{\mathfrak{N}})x}.$$

The idea is to find a vector v that maximizes the ratio R, i.e. $v \in \arg\max R(x)$. Given some threshold constant c, this vector v will give us a classification criteria: if $v^T\hat{d}_i > c$, we guess that $\hat{d}_i$ belongs to class $\mathfrak{P}$ and otherwise class $\mathfrak{N}$. In fact we may choose

$$v = (\Sigma_{\mathfrak{P}} + \Sigma_{\mathfrak{N}})^{-1}(m_{\mathfrak{P}} - m_{\mathfrak{N}})$$

and set $c = \frac{1}{2}v^T(m_{\mathfrak{P}} + m_{\mathfrak{N}})$. To avoid errors when $\Sigma_{\mathfrak{P}} + \Sigma_{\mathfrak{N}}$ is singular, we add a regularization term $\Sigma_{\mathfrak{P}} + \Sigma_{\mathfrak{N}} \to \Sigma_{\mathfrak{P}} + \Sigma_{\mathfrak{N}} + \lambda I$ for some small λ.*

3.2.4 Experiments

We consider the following classification tasks: $3 \mid h(d)$ vs $3 \nmid h(d)$, $5 \mid h(d)$ vs $5 \nmid h(d)$, $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$, $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$ and $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. Each task name follows the standard *positive* vs *negative*.

For each task we train both models on $\mathcal{D}_{\mathcal{T}_B^d}$, $\mathcal{D}_{\mathcal{T}_C^d}$ and $\mathcal{D}_{\mathcal{T}_D^d}$. The LDA is trained on $\mathcal{D}_{\mathcal{T}_A^d}$ as well. For the MLP $\mathscr{M}_\theta$, we train for 1500 epochs with the ADAM optimizer with hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use

*We use $\lambda = 1\text{e}{-4}$.

a constant learning rate $\texttt{lr} = 1\text{e}{-6}$. In regards to training and test split, we apply 5-fold cross-validation on a total dataset of $5\text{e}4$ data points such that each fold has equally many positive labels as negative labels. This establishes an elemental baseline classification accuracy at 0.5.

As mentioned, the MLP outputs the probability that the given input has positive label. Naturally, before evaluating the classification accuracy of $\mathscr{M}_\theta$, we need to convert these probabilities into binary classifications. We do this by employing a threshold at 0.5:

$$\text{pred}(d_i) \coloneqq \begin{cases} 1 \text{ if } (\mathscr{M}_\theta \circ \mathcal{T}^d)(d_i) \geq 0.5 \\ 0 \text{ otherwise.} \end{cases}$$

Let $\mathfrak{F}_j = \mathfrak{P}_j \cup \mathfrak{N}_j$ ($j \in \{1, 2, 3, 4, 5\}$) denote a test fold in the context of some classification task. For any model $\mathscr{M}$ (linear or not) with corresponding predictions $\text{pred}(d_i)$ for all $d_i \in \mathfrak{F}_j$, we define the accuracy (ACC), true positive rate (TPR) and true negative rate (TNR) as

$$\begin{aligned} \text{ACC} &\coloneqq \frac{\sum_{d_i \in \mathfrak{P}_j} \mathbb{1}_{\{\text{pred}(d_i) = y_{d_i}\}} + \sum_{d_i \in \mathfrak{N}_j} \mathbb{1}_{\{\text{pred}(d_i) = y_{d_i}\}}}{|\mathfrak{F}_j|}, \\ \text{TPR} &\coloneqq \frac{\sum_{d_i \in \mathfrak{P}_j} \mathbb{1}_{\{\text{pred}(d_i) = y_{d_i}\}}}{|\mathfrak{P}_j|} \text{ and} \\ \text{TNR} &\coloneqq \frac{\sum_{d_i \in \mathfrak{N}_j} \mathbb{1}_{\{\text{pred}(d_i) = y_{d_i}\}}}{|\mathfrak{N}_j|}. \end{aligned}$$

Chapter 4

Results

This chapter summarizes our main findings. Regarding the reviewed machine learning models, we make the important remark that all reported evaluation metrics as well as all data from the related figures concerns the corresponding *test* datasets — we do not show any results related to the *training* phases. In other words, we only report results on unseen data.

For brevity, we make use of the following notation: if a model (structure) $\mathscr{M}$ was trained on a feature space $\mathcal{D}$ in the context of some prediction task, we merge this into a unified evaluable model denoted as $[\mathscr{M}, \mathcal{D}]$.

4.1 Predicting $\mathcal{F}(h)$

This section presents the results obtained from conducting the experiments described in section 3.1.

In tables 4.1 and 4.2 we present evaluation metrics, for all models discussed in section 3.1, corresponding to the interpolation and extrapolation experiment, respectively. Figures 4.1, 4.2, 4.9 and 4.10 show the test loss for all models as a function of the number of training epochs. Figures 4.3, 4.5, 4.7, 4.11, 4.13, and 4.15 display histograms of the scaled test errors $r_{\text{model}}(h)$, which can be compared to the histograms of figure 3.1. Lastly, figures 4.4, 4.6, 4.8, 4.12, 4.14, and 4.16 show feature importance scores in descending order.

4.2 Predicting Class Numbers

This section presents results obtained from conducting the experiments described in section 3.2.

Table 4.1: (Interpolation experiment.) This table presents evaluation metrics for different models on the regression task of predicting $\mathcal{F}(h)$. The MSE is computed over the test dataset, and the statistics, μ and σ, are the sample mean and standard deviation of the scaled test errors $r_{\text{model}}(h)$. The reported results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}5$ data points.

	MSE/1e6	μ	σ
pred_0	535.3 ± 1.3	-25.364 ± 0.012	9.375 ± 0.009
pred_1	5.322 ± 0.021	2.209 ± 0.006	2.369 ± 0.005
pred_2	4.118 ± 0.013	0.140 ± 0.005	2.524 ± 0.006
pred_3	4.019 ± 0.014	0.268 ± 0.005	2.511 ± 0.006
pred_4	4.027 ± 0.014	0.257 ± 0.005	2.512 ± 0.006
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$	231.7 ± 6.6	-0.13 ± 0.38	18.60 ± 0.54
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$	0.832 ± 0.063	0.012 ± 0.031	1.56 ± 0.14
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$	0.76 ± 0.15	-0.005 ± 0.034	1.44 ± 0.20
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$	224.7 ± 3.3	0.06 ± 0.42	17.91 ± 0.16
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$	0.739 ± 0.038	0.024 ± 0.039	1.316 ± 0.028
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$	0.84 ± 0.15	-0.022 ± 0.055	1.46 ± 0.14

Table 4.2: (Extrapolation experiment.) This table presents evaluation metrics for different models on the regression task of predicting $\mathcal{F}(h)$. The MSE is computed over the test dataset, and the statistics, μ and σ, represent the sample mean and standard deviation of the scaled test errors $r_{\text{model}}(h)$.

	MSE/1e6	μ	σ
pred_0	1217.5	-34.1	6.83
pred_1	10.761	2.79	2.50
pred_2	7.9220	0.145	2.77
pred_3	7.7223	0.300	2.75
pred_4	7.7364	0.287	2.75
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$	832.51	-2.26	29.0
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$	75.452	-6.10	7.92
$[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$	112.53	-7.40	9.29
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$	1967.5	-0.576	45.7
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$	204.24	-10.2	12.9
$[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$	54.568	-4.18	6.85

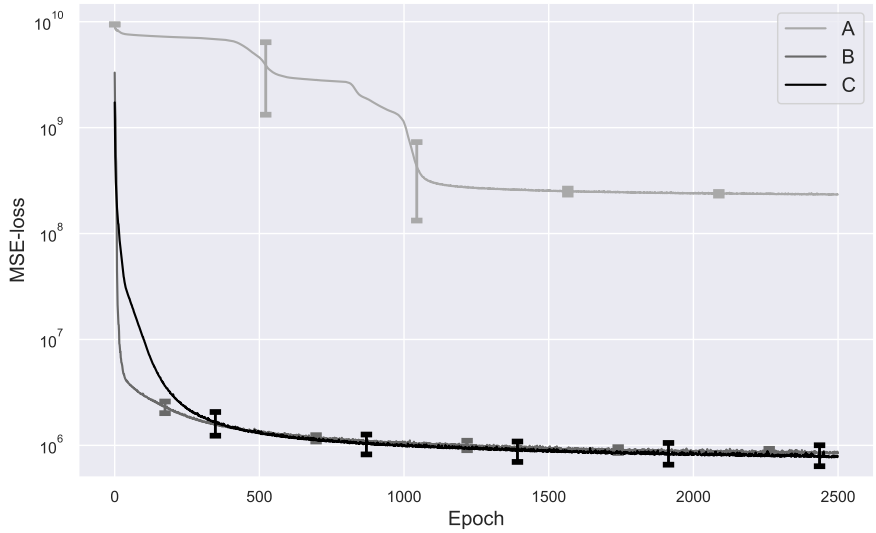


Figure 4.1: (Interpolation experiment.) Mean test loss for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}^h_{A,B,C}}]$. The error bars indicate one standard deviation over five folds of cross-validation.

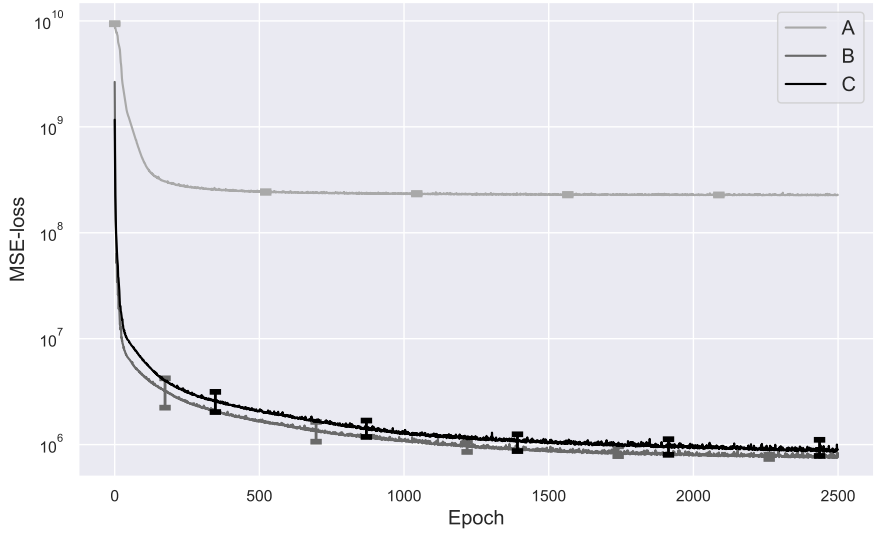
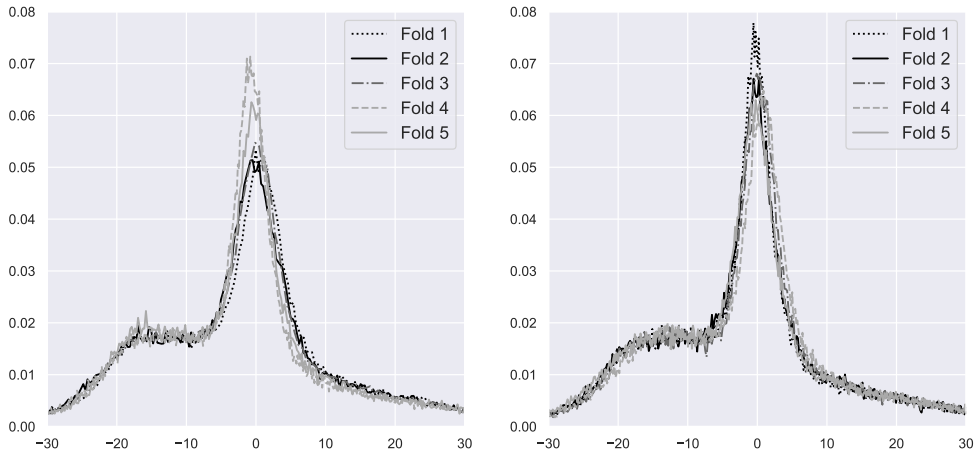


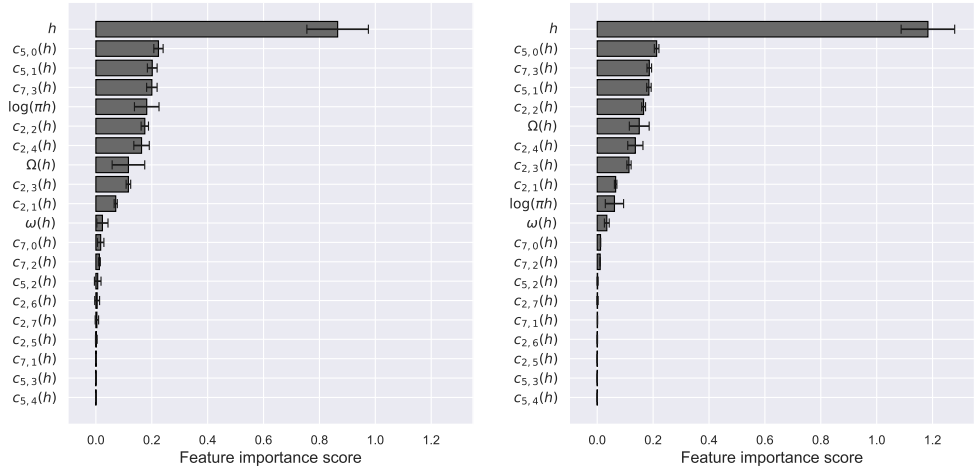
Figure 4.2: (Interpolation experiment.) Mean test loss for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}^h_{A,B,C}}]$. The error bars indicate one standard deviation over five folds of cross-validation.

Tables 4.3 – 4.7 contain evaluation metrics, for all models discussed in section 3.2, corresponding to the classification tasks that were presented in 3.2.4. Figures 4.17, 4.20, 4.23, 4.26 and 4.29 show the test loss, for all deep



(a) Error histograms for $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$.

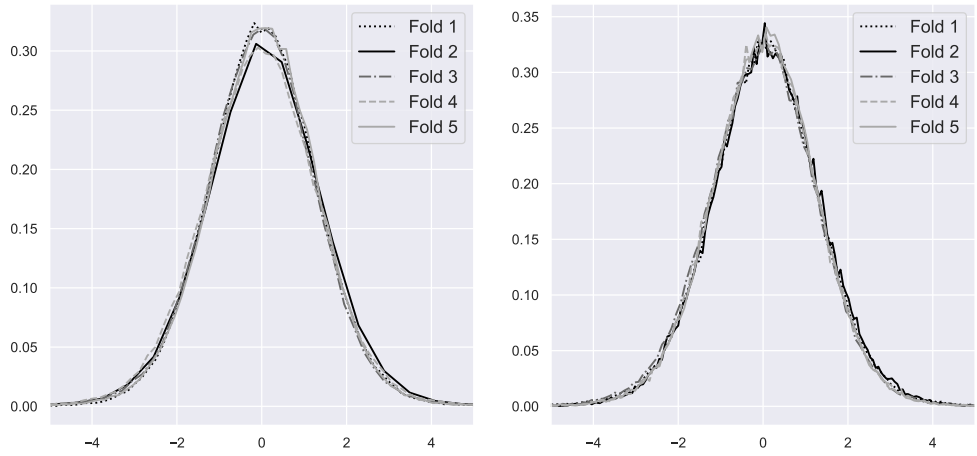
Figure 4.3: (Interpolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ for all folds $k = 1, \ldots, 5$.



(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$.

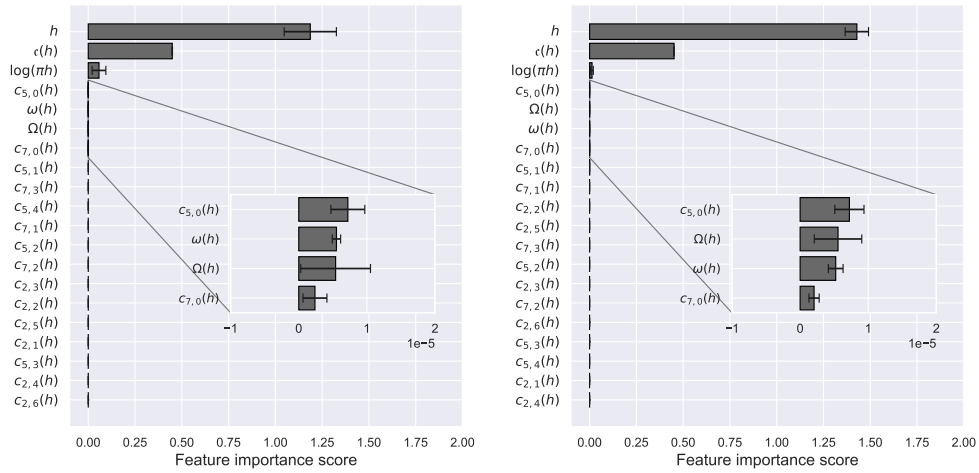
Figure 4.4: (Interpolation experiment.) Mean feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$. The error bars indicate one standard deviation over five folds of cross-validation.

learning models as a function of the number of training epochs, corresponding to each task. Similarly, figures 4.18, 4.21, 4.24, 4.27 and 4.30 show the test accuracy for all DL models and all tasks, as a function of the number of training epochs. For each task, we select the two DL models with highest performance



(a) Error histograms for $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_B^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_B^h}]$.

Figure 4.5: (Interpolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_B^h}]$ and $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_B^h}]$ for all folds $k = 1, \ldots, 5$.



(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$.

Figure 4.6: (Interpolation experiment.) Mean feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$. The error bars indicate one standard deviation over five folds of cross-validation.

and, for these models, display the top 20 most important features in descending order — see figures 4.19, 4.22, 4.25, 4.28 and 4.31.

Table 4.3: Evaluation metrics for different models on the classification task $3 \mid h(d)$ vs $3 \nmid h(d)$. The results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}4$ data points.

	[LDA,$\mathcal{D}_{\mathcal{T}_A^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_B^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_C^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_D^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_B^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_C^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_D^d}$]
ACC	$.501 \pm .004$	$.502 \pm .004$	$.505 \pm .007$	$.503 \pm .003$	$.506 \pm .002$	$.506 \pm .002$	$.508 \pm .003$
TPR	$.508 \pm .005$	$.499 \pm .006$	$.502 \pm .010$	$.498 \pm .001$	$.590 \pm .099$	$.565 \pm .111$	$.511 \pm .035$
TNR	$.493 \pm .008$	$.505 \pm .011$	$.507 \pm .005$	$.507 \pm .008$	$.423 \pm .097$	$.446 \pm .111$	$.506 \pm .035$

Table 4.4: Evaluation metrics for different models on the classification task $5 \mid h(d)$ vs $5 \nmid h(d)$. The results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}4$ data points.

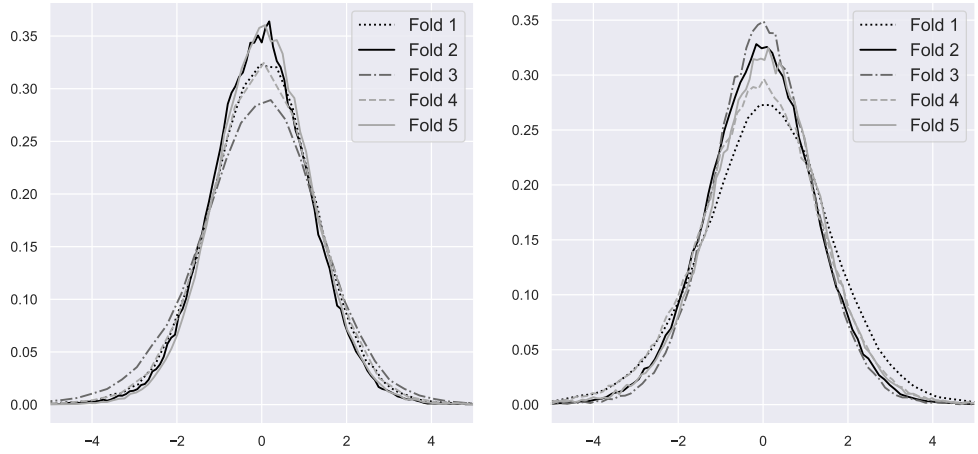
	[LDA,$\mathcal{D}_{\mathcal{T}_A^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_B^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_C^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_D^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_B^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_C^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_D^d}$]
ACC	$.501 \pm .004$	$.503 \pm .002$	$.502 \pm .004$	$.501 \pm .006$	$.512 \pm .001$	$.509 \pm .002$	$.505 \pm .002$
TPR	$.508 \pm .003$	$.504 \pm .005$	$.502 \pm .004$	$.500 \pm .011$	$.615 \pm .136$	$.567 \pm .067$	$.625 \pm .163$
TNR	$.493 \pm .005$	$.501 \pm .003$	$.502 \pm .010$	$.502 \pm .009$	$.408 \pm .138$	$.451 \pm .068$	$.385 \pm .165$

Table 4.5: Evaluation metrics for different models on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$. The results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}4$ data points.

	[LDA,$\mathcal{D}_{\mathcal{T}_A^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_B^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_C^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_D^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_B^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_C^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_D^d}$]
ACC	$.516 \pm .002$	$.528 \pm .005$	$.522 \pm .004$	$.523 \pm .003$	$.528 \pm .002$	$.519 \pm .005$	$.526 \pm .003$
TPR	$.524 \pm .005$	$.526 \pm .007$	$.525 \pm .006$	$.522 \pm .005$	$.495 \pm .030$	$.477 \pm .047$	$.506 \pm .008$
TNR	$.509 \pm .005$	$.530 \pm .006$	$.520 \pm .003$	$.524 \pm .005$	$.561 \pm .025$	$.562 \pm .046$	$.546 \pm .006$

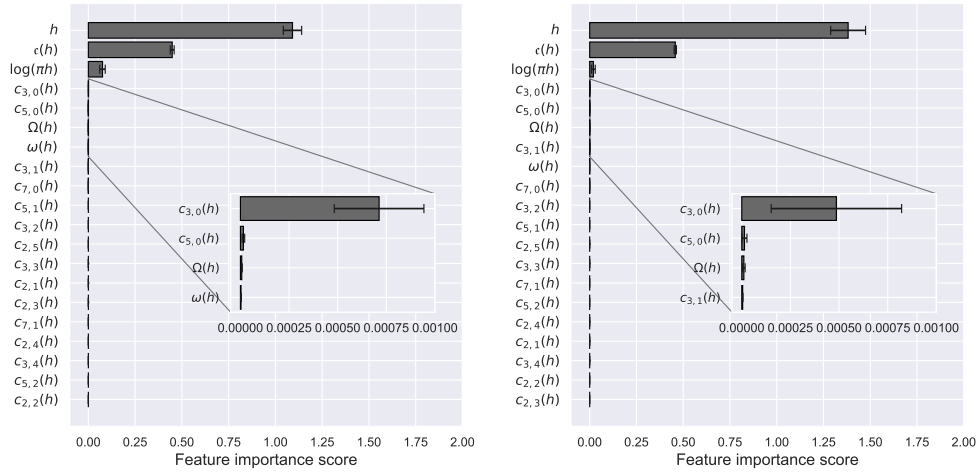
Table 4.6: Evaluation metrics for different models on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$. The results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}4$ data points.

	[LDA,$\mathcal{D}_{\mathcal{T}_A^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_B^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_C^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_D^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_B^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_C^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_D^d}$]
ACC	$.523 \pm .006$	$.536 \pm .003$	$.534 \pm .003$	$.534 \pm .004$	$.535 \pm .003$	$.532 \pm .005$	$.534 \pm .004$
TPR	$.528 \pm .005$	$.529 \pm .003$	$.534 \pm .006$	$.532 \pm .009$	$.512 \pm .010$	$.516 \pm .010$	$.521 \pm .008$
TNR	$.517 \pm .011$	$.544 \pm .006$	$.534 \pm .007$	$.536 \pm .003$	$.558 \pm .011$	$.548 \pm .006$	$.547 \pm .005$



(a) Error histograms for $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_C^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_C^h}]$.

Figure 4.7: (Interpolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_C^h}]$ and $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_C^h}]$ for all folds $k = 1, \ldots, 5$.



(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$.

Figure 4.8: (Interpolation experiment.) Mean feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$. The error bars indicate one standard deviation over five folds of cross-validation.

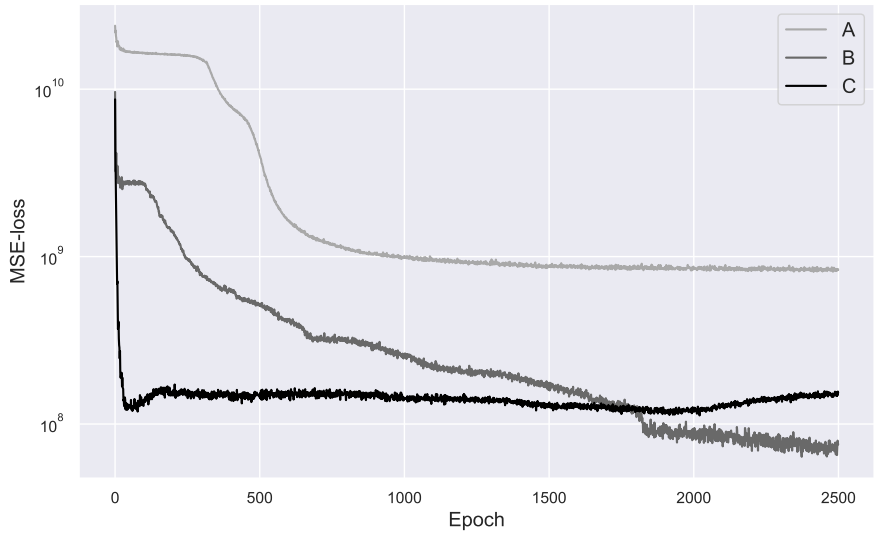


Figure 4.9: (Extrapolation experiment.) Test loss for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}^h_{A,B,C}}]$.

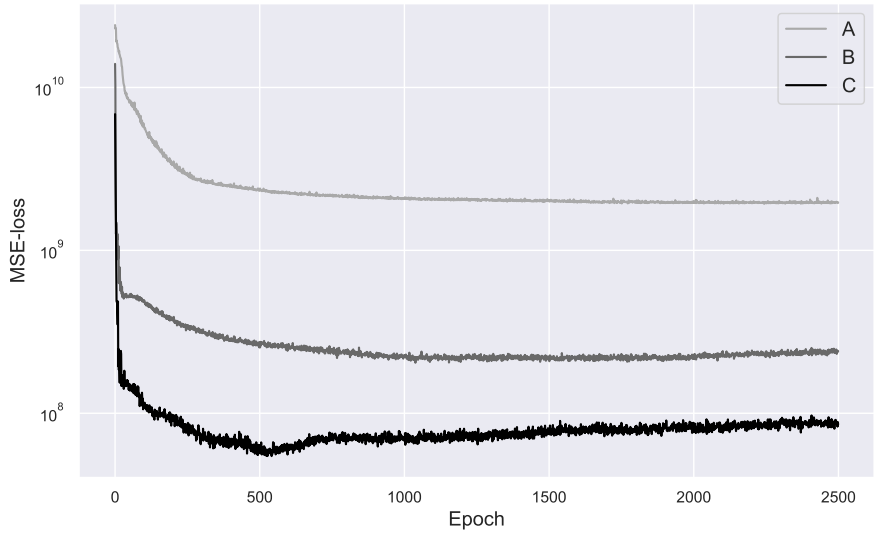
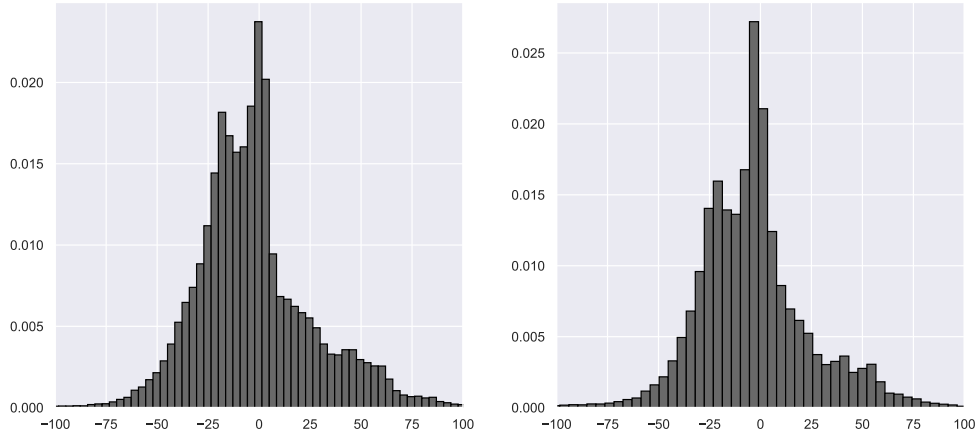
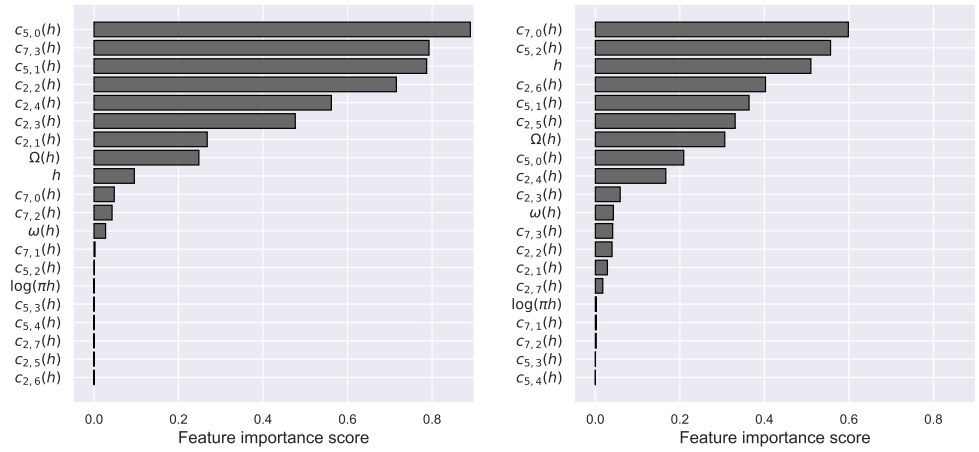


Figure 4.10: (Extrapolation experiment.) Test loss for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}^h_{A,B,C}}]$.



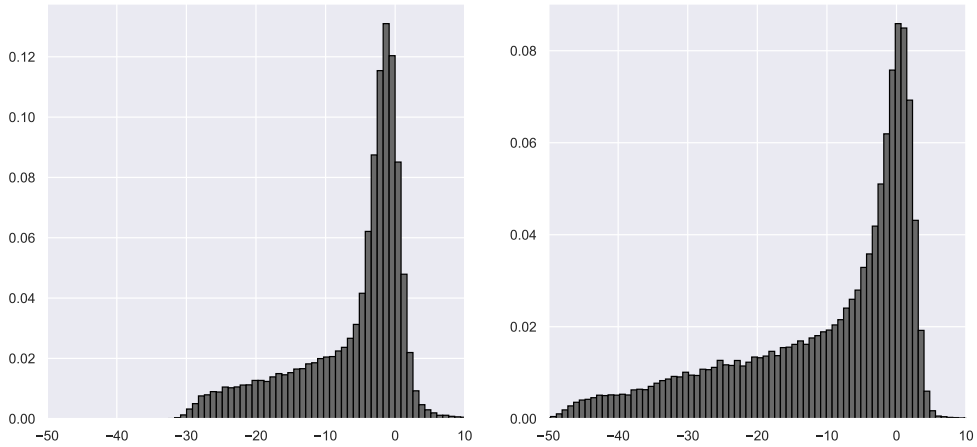
(a) Error histograms for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$.

Figure 4.11: (Extrapolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$.



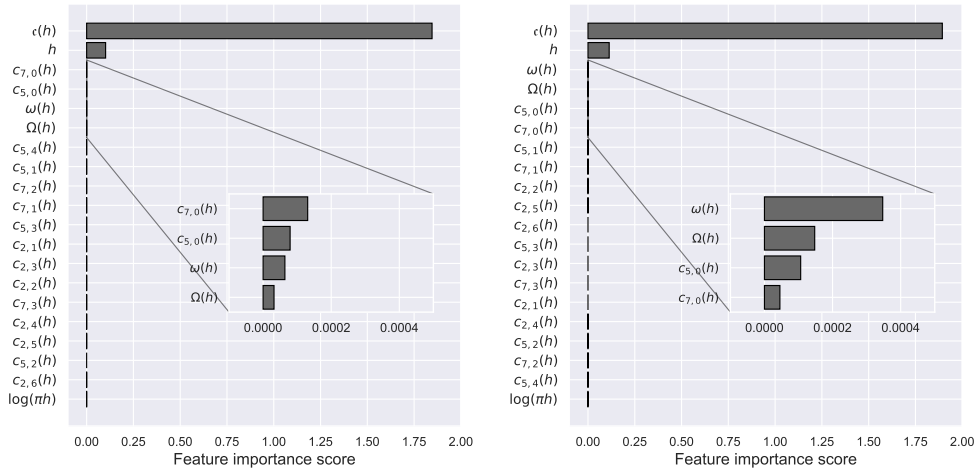
(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$.

Figure 4.12: (Extrapolation experiment.) Feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_A^h}]$.



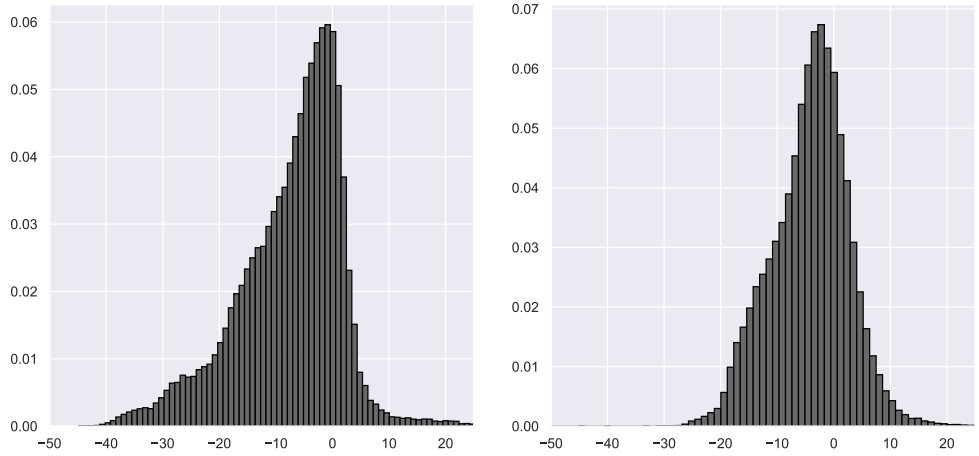
(a) Error histograms for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$.

Figure 4.13: (Extrapolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$.



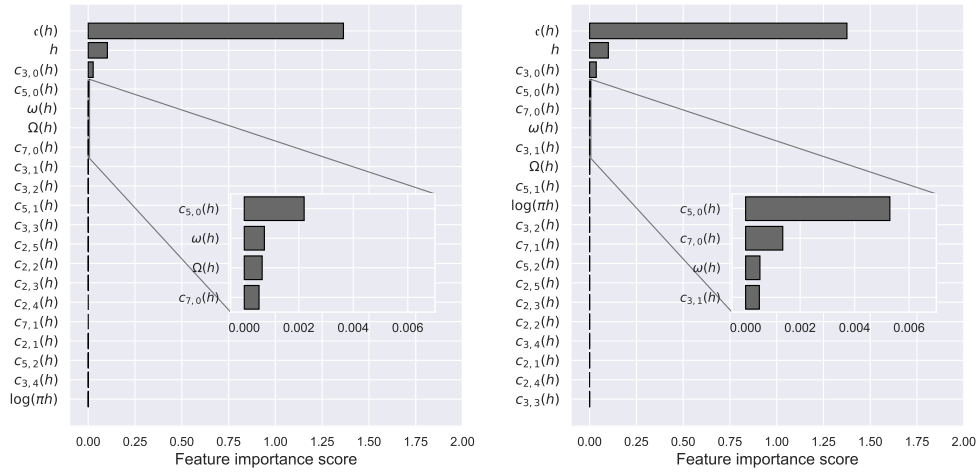
(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$.

Figure 4.14: (Extrapolation experiment.) Feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_B^h}]$.



(a) Error histograms for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$. (b) Error histograms for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$.

Figure 4.15: (Extrapolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$.



(a) Importance scores for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$. (b) Importance scores for $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$.

Figure 4.16: (Extrapolation experiment.) Feature importance scores in descending order for the models $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_C^h}]$ and $[\mathscr{M}_{\theta_2}, \mathcal{D}_{\mathcal{T}_C^h}]$.

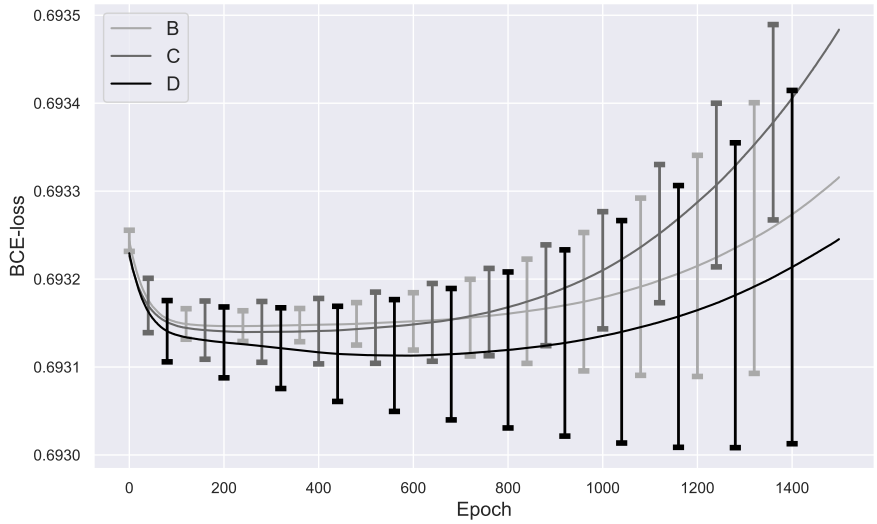


Figure 4.17: Mean test loss for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $3 \mid h(d)$ vs $3 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

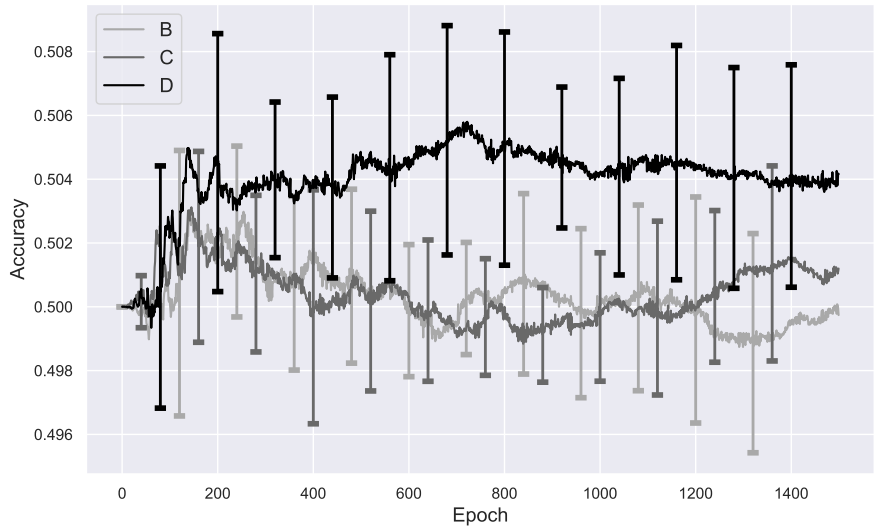
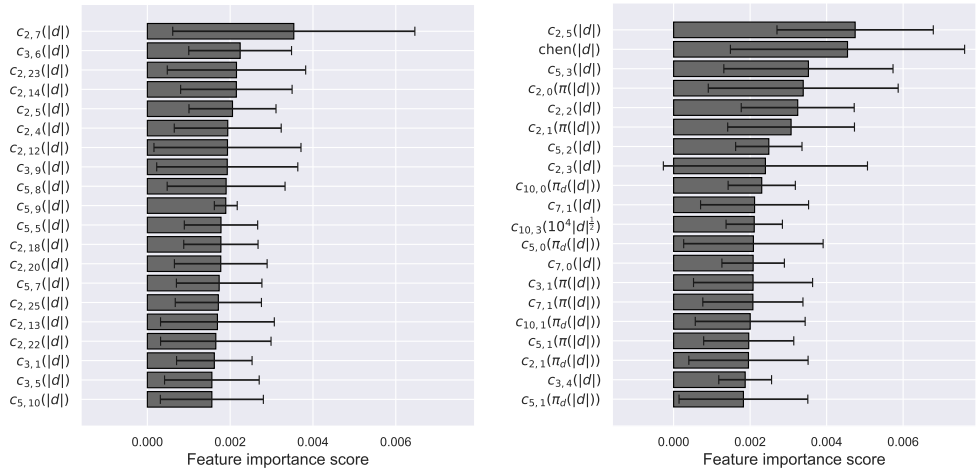


Figure 4.18: Mean test accuracy for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $3 \mid h(d)$ vs $3 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.



(a) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_B^d}]$. (b) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_D^d}]$.

Figure 4.19: Top 20 mean feature importance scores in descending order for the two best DL models — according to table 4.3 — on the classification task $3 \mid h(d)$ vs $3 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

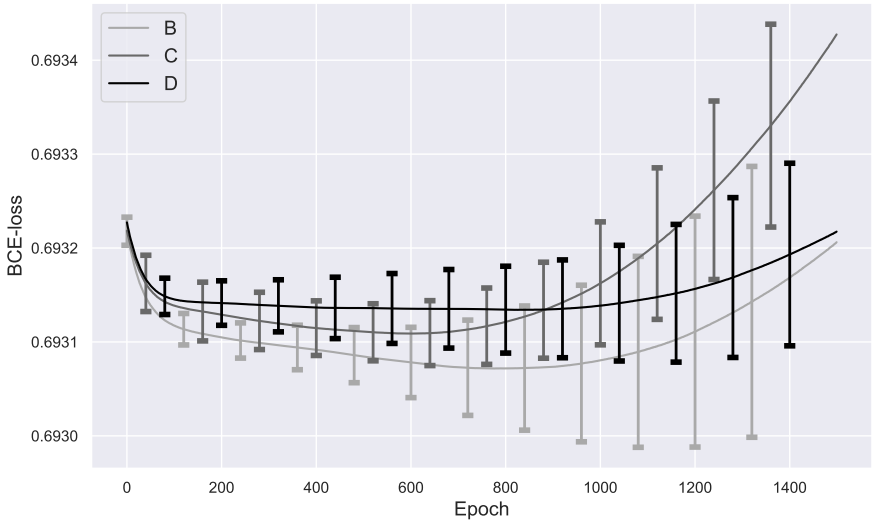


Figure 4.20: Mean test loss for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_{B,C,D}^d}]$ on the classification task $5 \mid h(d)$ vs $5 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

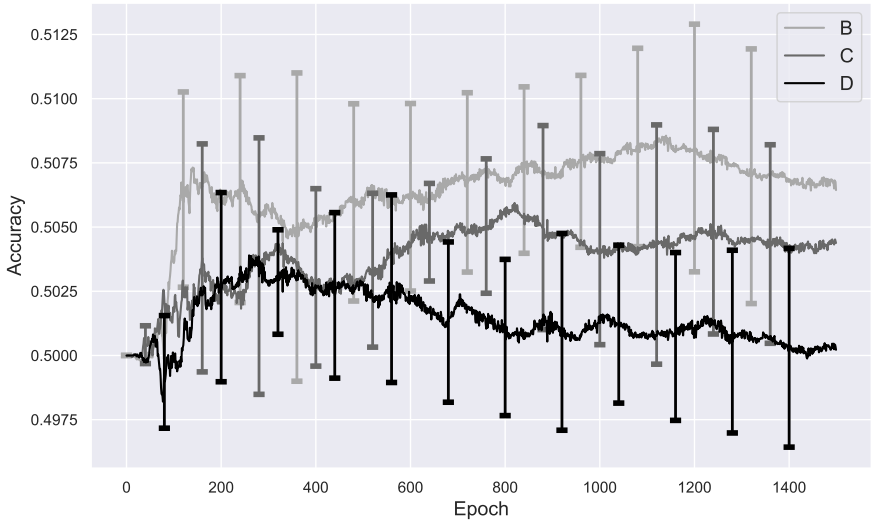
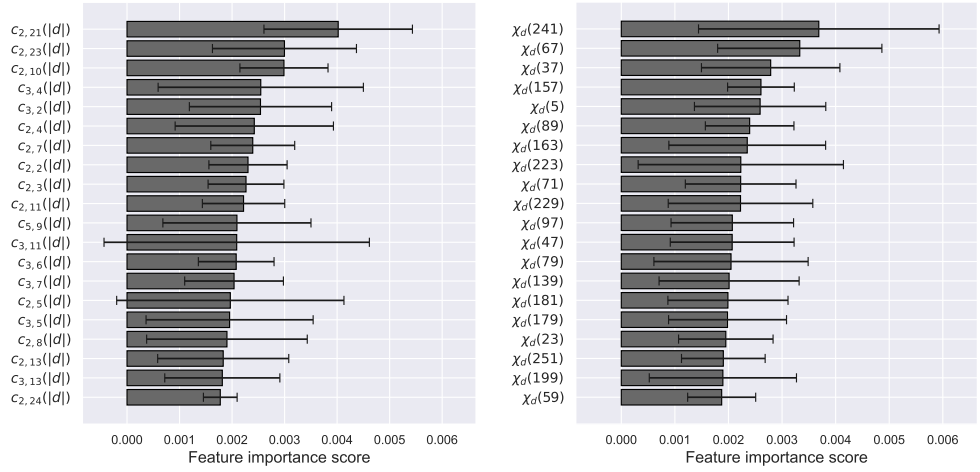


Figure 4.21: Mean test accuracy for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $5 \mid h(d)$ vs $5 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.



(a) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_B}]$.

(b) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_C}]$.

Figure 4.22: Top 20 mean feature importance scores in descending order for the two best DL models — according to table 4.4 — on the classification task $5 \mid h(d)$ vs $5 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

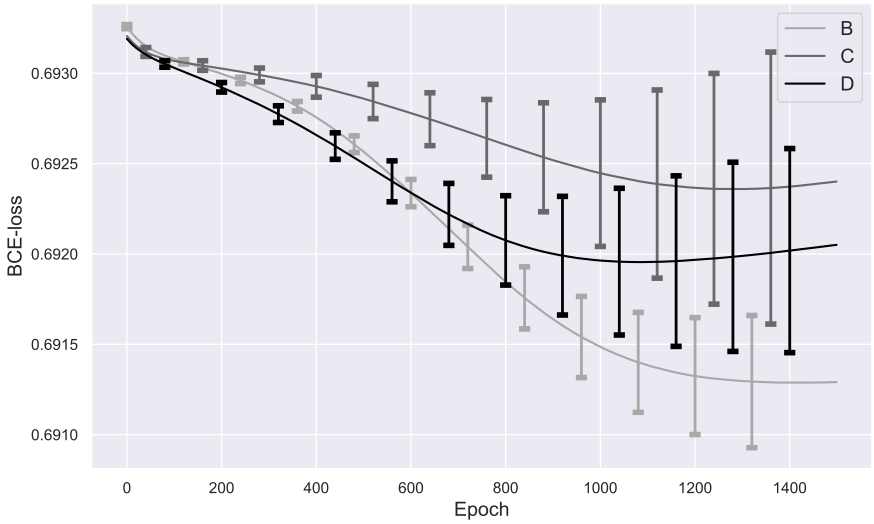


Figure 4.23: Mean test loss for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$. The error bars indicate one standard deviation over five folds of cross-validation.

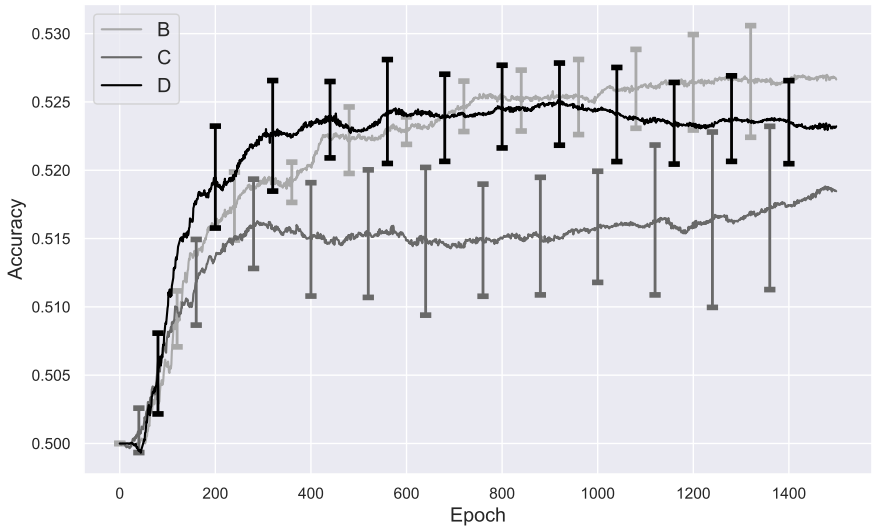
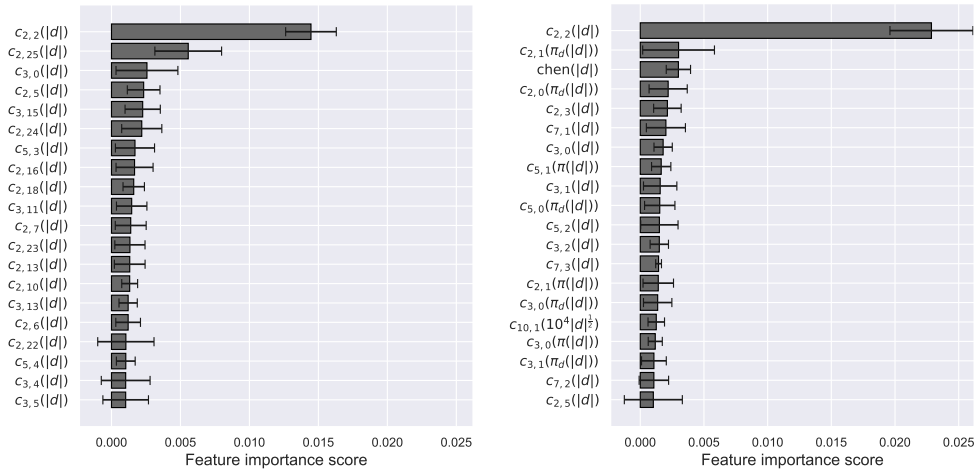


Figure 4.24: Mean test accuracy for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$. The error bars indicate one standard deviation over five folds of cross-validation.



(a) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_B^d}]$. (b) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_D^d}]$.

Figure 4.25: Top 20 mean feature importance scores in descending order for the two best DL models — according to table 4.5 — on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$. The error bars indicate one standard deviation over five folds of cross-validation.

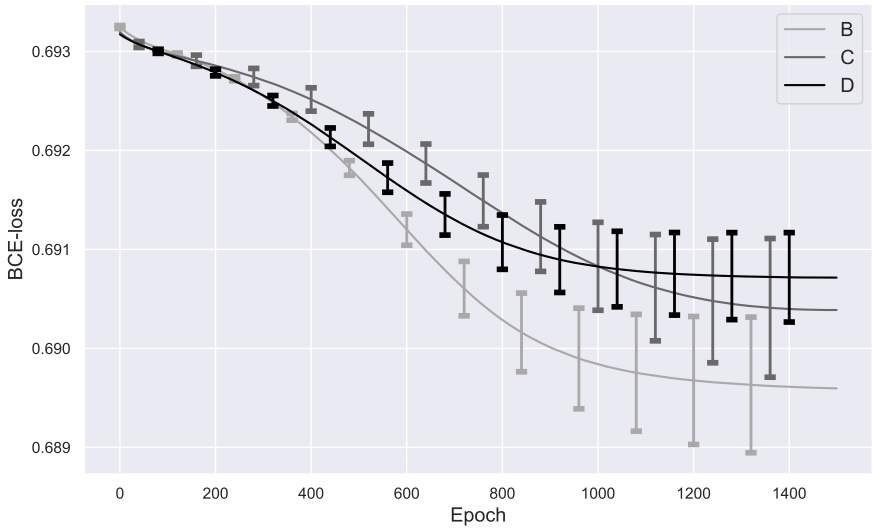


Figure 4.26: Mean test loss for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_{B,C,D}^d}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$. The error bars indicate one standard deviation over five folds of cross-validation.

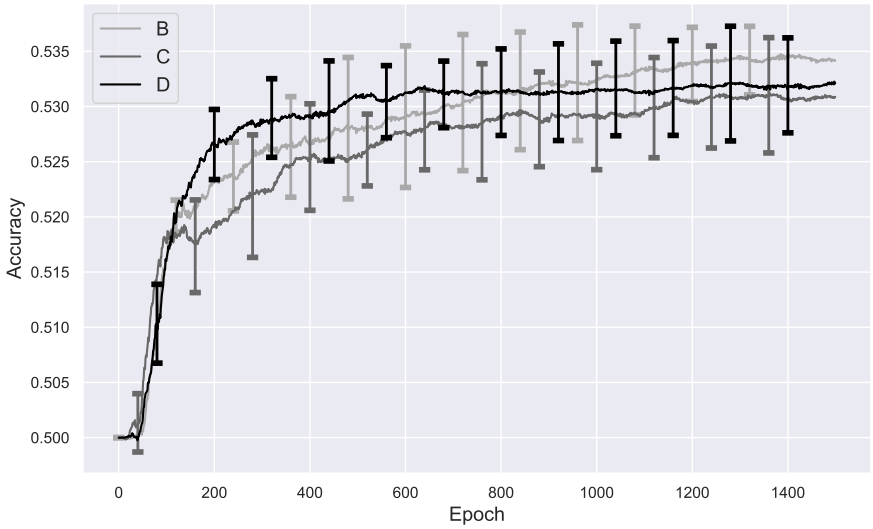
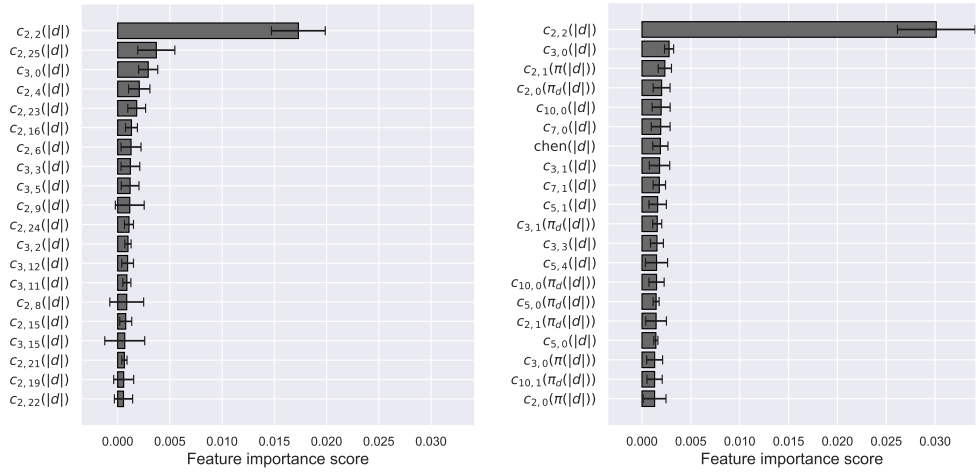


Figure 4.27: Mean test accuracy for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$. The error bars indicate one standard deviation over five folds of cross-validation.



(a) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_B}]$.

(b) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_D}]$.

Figure 4.28: Top 20 mean feature importance scores in descending order for the two best DL models — according to table 4.6 — on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$. The error bars indicate one standard deviation over five folds of cross-validation.

Table 4.7: Evaluation metrics for different models on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. The results represent the mean $\pm$ standard deviation over five folds of cross-validation, performed on a total set of $5\text{e}4$ data points.

	[LDA,$\mathcal{D}_{\mathcal{T}_A^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_B^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_C^d}$]	[LDA,$\mathcal{D}_{\mathcal{T}_D^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_B^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_C^d}$]	[$\mathscr{M}_\theta$,$\mathcal{D}_{\mathcal{T}_D^d}$]
ACC	$.529 \pm .004$	$.551 \pm .001$	$.548 \pm .005$	$.544 \pm .004$	$.551 \pm .001$	$.546 \pm .005$	$.545 \pm .002$
TPR	$.533 \pm .005$	$.540 \pm .007$	$.547 \pm .010$	$.544 \pm .008$	$.536 \pm .007$	$.532 \pm .009$	$.522 \pm .008$
TNR	$.525 \pm .008$	$.563 \pm .009$	$.549 \pm .003$	$.543 \pm .006$	$.566 \pm .007$	$.559 \pm .005$	$.568 \pm .007$

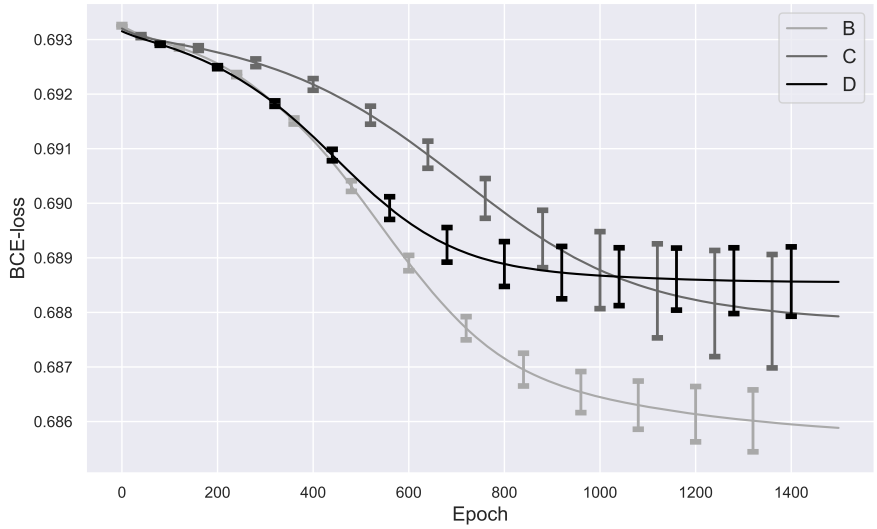


Figure 4.29: Mean test loss for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_{B,C,D}^d}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. The error bars indicate one standard deviation over five folds of cross-validation.

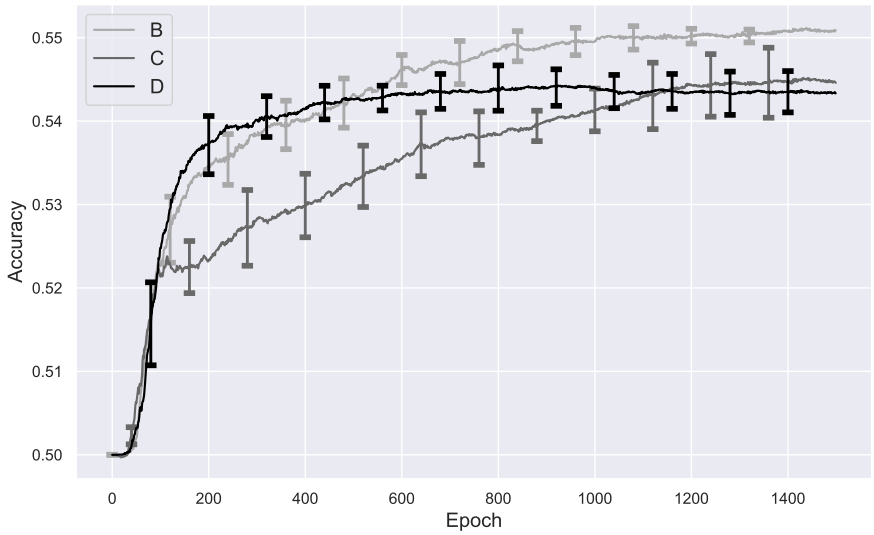
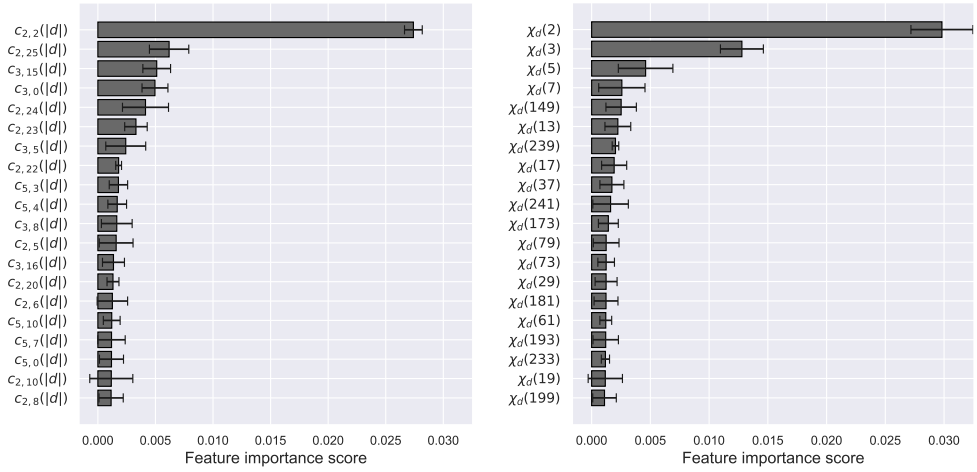


Figure 4.30: Mean test accuracy for the models $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_{B,C,D}}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. The error bars indicate one standard deviation over five folds of cross-validation.



(a) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_B}]$.

(b) Importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}^d_C}]$.

Figure 4.31: Top 20 mean feature importance scores in descending order for the two best DL models — according to table 4.7 — on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. The error bars indicate one standard deviation over five folds of cross-validation.

Chapter 5

Discussion

We discuss the results from chapter 4 and we divide the discussion into two parts. The first part focuses on the results from section 4.1 while the other part concerns the results from section 4.2.

5.1 Predicting $\mathcal{F}(h)$

When it comes to interpolating $\mathcal{F}(h)$, the results from section 4.1 show that the deep learning models that have explicit knowledge of $\mathfrak{c}(h)$ outperform the benchmark models. However, the performance of the models does not appear to improve significantly when provided with $h \mod 3$ in addition to $\mathfrak{c}(h)$. This suggests that $\mathfrak{c}(h)$ contains enough divisibilty-information of h for the models to extract three-divisibilty on their own. Moreover, it is interesting to analyze the feature importance scores. We see that h (and $\log(\pi h)$ for interpolation) and, when provided, $\mathfrak{c}(h)$ are dominant features in regards to importance scores. Generally speaking, we also see that the models make use of knowing 3-, 5- and 7-divisibilty of h and these features seem to have decreasing rank in terms of importance scores. Thus, having explicit knowledge of which small prime numbers divide h appears to be important for accurate predictions. However, since the results for the models that do not know $\mathfrak{c}(h)$ are poor, it is evidently not enough to only have $h \mod 5$ and $h \mod 7$ etc. as inputs. We also note that $\Omega(h)$ and $\omega(h)$ consistently appear as relatively important features.

Of course, we expect the performance of the models in the extrapolation experiment to be worse compared to interpolation. This is because the models in the former case need to predict outputs based on values of h that are beyond the observed training data. Neural networks and MLPs in particular tend

to struggle when it comes to this kind of extrapolation and generalizing to examples far from the training data [23]. Indeed, figures 4.12, 4.14 and 4.16 show that the size of the input, h, is a significant feature.

In regards to the extrapolation experiment, it is interesting to compare the shapes of the $r_{\text{model}}(h)$-distributions displayed in figures 4.11, 4.13 and 4.15. Recall that the feature space $\mathcal{D}_{\mathcal{T}_A^h}$ does not contain $\mathfrak{c}(h)$, yet the histograms in 4.11a and 4.11b resemble the distributions in figure 3.1 in that there are two distinct peaks. The histograms in 4.13a and 4.13b have heavy tails towards the left, which indicate that most of the prediction error is due to overshooting (predictions larger than the actual) rather than undershooting (predictions smaller than the actual). This behaviour seems to subside when $h \mod 3$ is included in the feature vector together with $\mathfrak{c}(h)$ — see figures 4.15a and 4.15b, and note that their distributions do not have as distinct tails as in 4.13a and 4.13b. Since the test loss for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ has not converged after 2500 epochs and is still decreasing (see figure 4.9), we let it run for another 2500 epochs; the results are presented in figures C.3 and C.4. For this extended experiment, we recorded the metrics MSE $\approx 44.785 \cdot 10^6$, $\mu \approx -3.252$ and $\sigma \approx 6.746$, which is better than all DL performances in table 4.2, but still does not beat the analytical benchmarks.

As a final note we remark that the loss functions for the model (structure) $\mathscr{M}_{\theta_2}$ tend to converge smoother and faster for all feature spaces compared to $\mathscr{M}_{\theta_1}$ — compare figure 4.2 and 4.10 with 4.1 and 4.9.

5.2 Predicting Class Numbers

Evidently, our models struggle with separating discriminants based on explicit 3- or 5-divisibility of the class number. Yet, when relaxing the divisibility requirements and instead considering the number of prime factors of $h(d)$, the models seem to perform better. In fact, our results indicate that the separation of discriminants d based on $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq n$ gets successively easier when n grows. What is interesting about this is that when we look at figures 4.25, 4.28 and 4.31 we see that essentially only one feature is responsible for these results — namely $c_{2,2}(|d|)$. Below we discuss the reason for the $c_{2,2}(|d|)$-imbalance in this context.

Given any negative fundamental prime discriminant $d = -(\ldots + c_{2,4}(|d|)2^4 + c_{2,3}(|d|)2^3 + c_{2,2}(|d|)2^2 + 2 + 1)$, the coefficients $c_{2,i}(|d|)$ are either zero or one. If $c_{2,2}(|d|) = 0$ we have $d \equiv 5 \mod 8$ and if $c_{2,2}(|d|) = 1$ we have $d \equiv 1 \mod 8$. Intuitively for large discriminants (say $|d| > 10^7$), we should have $P(c_{2,i}(|d|) = 1) \approx 1/2$ for $i = 2, 3, 4, 5, \ldots$ up to some

threshold. Indeed, this is supported by table 5.1. However, when we condition the probabilities on $\Omega(h(d))$, deviant patterns emerge for $c_{2,2}(|d|)$ (see table 5.2). Though, these results can be explained by the class number formula 1.2.1. Recall that

$$h(d) = \frac{\sqrt{|d|}}{\pi} L(1, \chi_d) = \frac{\sqrt{|d|}}{\pi} \prod_p \left(1 - \frac{1}{p}\chi_d(p)\right)^{-1}$$

and $\chi_d(p) \in \{-1, 1\} \forall p \neq |d|$. From this it is clear that the value of $\chi_d(p)$ for $p = 2$ has the greatest impact on the size of $h(d)$. Moreover we have

$$\left(\frac{d}{2}\right) = \begin{cases} 1 \text{ if } d \equiv 1 \mod 8, \\ -1 \text{ if } d \equiv 5 \mod 8, \end{cases}$$

which suggests that $h(d)$ is typically larger when $d \equiv 1 \mod 8$ compared to $d \equiv 5 \mod 8$. This, in turn, would explain why for instance the ratios (see table 5.2)

$$\frac{\#\{d : |d| \in \{10^m, 10^m + 5 \cdot 10^8\}, \Omega(h(d)) = 7, d \equiv 1 \mod 8\}}{\#\{d : |d| \in \{10^m, 10^m + 5 \cdot 10^8\}, \Omega(h(d)) = 7\}} > 1/2$$

for relatively small m. However, this effect naturally wears off when $|d| \to \infty$. When we consider prime class numbers, this relation is reversed; larger class numbers are primes with smaller probability.

For the sake of a more explicit, yet informal, argument in the case $h(d)$ is prime, let us fix our focus on some given range $\mathfrak{D} \subseteq \mathbb{R}_{\geq 0}$ for the values of $|d|$, and suppose we have

$$P(h(d_2) > h(d_1) \mid d_2 \equiv 1 \bmod 8, d_1 \equiv 5 \bmod 8, |d_1|, |d_2| \in \mathfrak{D}) = 1. \quad (5.1)$$

This implies the existence of a constant κ such that $|d| \in \mathfrak{D} \wedge h(|d|) < \kappa \Rightarrow d \equiv 5 \mod 8$ and $|d| \in \mathfrak{D} \wedge h(|d|) > \kappa \Rightarrow d \equiv 1 \mod 8$. From Bayes' theorem it follows that

$$P(d \equiv 1 \mod 8 \mid h(d) \text{ prime}, |d| \in \mathfrak{D}) = \frac{P(h(d) \text{ prime} \mid d \equiv 1 \mod 8, |d| \in \mathfrak{D}) P(d \equiv 1 \mod 8 \mid |d| \in \mathfrak{D})}{P(h(d) \text{ prime} \mid |d| \in \mathfrak{D})} \approx \frac{1}{2} \frac{P(h(d) \text{ prime} \mid d \equiv 1 \mod 8, |d| \in \mathfrak{D})}{P(h(d) \text{ prime} \mid |d| \in \mathfrak{D})}.$$

We make use of the following approximations

$$P(q < x \text{ prime}) \approx \frac{x}{\log(x)}/x = \frac{1}{\log(x)},$$
$$P(y < q < x \text{ prime}) \approx \frac{x\log(y) - y\log(x)}{(x-y)\log(x)\log(y)},$$

as well as the bounds

$$(1+o(1))(\pi/12e^{\gamma})\sqrt{|d|}/\log(\log(|d|)) < h(d) < \sqrt{|d|}\log(|d|)$$

(see [7, p. 908]). * Call the LHS $h(d)_{\min}$ and the RHS $h(d)_{\max}$. From this we get the following estimates:

$$P(h(d) \text{ prime} \mid |d| \in \mathfrak{D}) \approx P[h(-\inf\mathfrak{D})_{\min} < q < h(-\sup\mathfrak{D})_{\max} \text{ prime}],$$
$$P(h(d) \text{ prime} \mid d \equiv 1 \mod 8, |d| \in \mathfrak{D}) \approx P[\kappa < q < h(-\sup\mathfrak{D})_{\max} \text{ prime}].$$

If we now suppose that $\kappa = (h(-\sup\mathfrak{D})_{\max} - h(-\inf\mathfrak{D})_{\min})/2$ and we set for instance $\mathfrak{D} = [10^{12}, 10^{12} + 5\cdot 10^8]$, we get

$$P(h(d) \text{ prime} \mid |d| \in \mathfrak{D}) \approx 0.058305,$$
$$P(h(d) \text{ prime} \mid d \equiv 1 \mod 8, |d| \in \mathfrak{D}) \approx 0.055903,$$

which yields

$$P(d \equiv 1 \mod 8 \mid h(d) \text{ prime}, |d| \in \mathfrak{D}) \approx 0.47940.$$

This agrees quite well with table 5.2. When we compute the actual probability 5.1, corresponding to this example, we get ≈ 0.9535 instead of 1.0. If we perform the estimation on $\mathfrak{D} = [10^{15}, 10^{15} + 5\cdot 10^8]$ instead, we get

$$P(d \equiv 1 \mod 8 \mid h(d) \text{ prime}, |d| \in \mathfrak{D}) \approx 0.48306$$

and corresponding 5.1-probability ≈ 0.9535.

We remark that this explanatory model does not rest on rigorous mathematical foundations, but rather aims to give the reader a rough idea of what could be the cause of some of the results in table 5.2.

Let us use the above explanation to argue why we can expect the best models from table 4.5 to reach an accuracy of ≈ 0.53 based only on the values of $c_{2,2}(|d|)$. With $\mathfrak{D} = [10^6, 65\cdot 10^6]$ (which corresponds to the discriminant

*$\gamma \approx 0.57721566490153286060651209008240243$ is the Euler-Mascheroni constant.

range for our experiments) we get, by following the above explanation model,

$$P(d \equiv 1 \mod 8 \mid h(d) \text{ prime}, |d| \in \mathfrak{D}) \approx 0.47.$$

Thus, a model that predicts $h(d)$ as being prime for all discrimiants with $d \equiv 5 \mod 8$ and as not being prime for all $d \equiv 1 \mod 8$ would get an accuracy of around $53/100 = 0.53$.

We remark that the class number formula indicates that $\chi_d(3)$ also has significant impact on the size of $h(d)$, and should therefore also be a useful feature to the models. Recall that

$$\chi_d(3) = \begin{cases} 1 \text{ if there is an integer } n \in (0,3) : n^2 = d \mod 3, \\ -1 \text{ otherwise} \end{cases}$$

and note that this means that $d \equiv 1 \mod 3 \Leftrightarrow \chi_d(3) = 1$ and $d \equiv 2 \mod 3 \Leftrightarrow \chi_d(3) = -1$. Thus the value of $\chi_d(3)$ is indirectly given to the models through the feature $c_{3,0}(|d|)$, which we indeed can see, from figures 4.25, 4.28 and 4.31, is of great importance. However, this impact is not as noticeable as for $c_{2,2}(|d|)$. Figure 4.31b clearly demonstrates that $\mathscr{M}_\theta$ is using $\chi_d(p)$ for $p = 2, 3, 5, 7$ to (likely) estimate the size of $h(d)$ and thereby get a decent guess on $\Omega(h(d))$.

We have discussed that the reason for the success of the models when it comes to separating discriminants based on $\Omega(h(d))$ is likely due to the presence of features that enable the models to approximate the size of $h(d)$. In turn, this means that the better the approximation of the size of $h(d)$, the higher the classification accuracy. This is probably why we see that $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_B^d}]$ consistently outperforms $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_C^d}]$ and $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_D^d}]$ — having $c_{2,2}(|d|)$ and $c_{3,0}(|d|)$ together with $c_{2,25}(|d|)$ is better than only having many $\chi_d(p)$.

Table 5.1: This table presents some of the quantities $10^4 \cdot [P(c_{2,i}(|d|) = 1 \mid |d| \in [10^m, 10^m + 5 \cdot 10^8]) - 1/2]$.

$m \backslash i$	2	3	4	5	6	7
8	0.148	0.293	-0.392	-0.699	0.011	0.497
9	0.327	0.258	-0.153	0.886	0.064	-0.525
10	-0.366	0.498	-0.191	-0.285	0.099	-1.675
11	0.278	0.723	-0.608	-0.804	0.411	-1.117
12	0.112	1.253	0.800	-0.993	-1.305	1.101
13	0.162	-0.929	0.374	0.047	-0.001	0.941
14	-0.804	-0.409	0.787	1.651	-0.956	-3.362

Table 5.2: This table presents some of the conditional probabilities $P(d \equiv 1 \mod 8 \mid \Omega(h(d)) = n, |d| \in [10^m, 10^m + 5 \cdot 10^8])$.

$m \backslash n$	1	2	3	4	5	6	7
8	0.470	0.487	0.507	0.529	0.550	0.580	0.603
9	0.472	0.487	0.505	0.524	0.542	0.566	0.593
10	0.473	0.487	0.503	0.520	0.536	0.554	0.574
11	0.476	0.488	0.502	0.515	0.529	0.542	0.559
12	0.479	0.488	0.500	0.513	0.524	0.538	0.549
13	0.481	0.489	0.499	0.512	0.521	0.530	0.541
14	0.482	0.489	0.499	0.508	0.519	0.528	0.532
15	0.483	0.490	0.498	0.507	0.515	0.525	0.529
16	0.484	0.491	0.498	0.506	0.515	0.520	0.525
17	0.486	0.491	0.498	0.505	0.511	0.518	0.523
18	0.487	0.492	0.497	0.504	0.511	0.515	0.523

Chapter 6

Conclusions and Future Work

We present conclusions to be drawn from the work as well as ideas for future work, among other things.

6.1 Conclusions

Regarding question 1 we conclude that for the input range $h \in [1, 10^6]$, elementary deep learning models show great potential in interpolating $\mathcal{F}(h)$. Yet, it seems more difficult for these models to extrapolate to larger data points. Whether we are concerned with interpolation or extrapolation, the feature $\mathfrak{c}(h)$ appears to be imperative for the relative success of the models.

As for question 2, there are weak indications that deep learning models outperform linear models when it comes to predicting 3- and 5-divisibility of a random class number based on discriminant information. Nevertheless, the results from section 4.2 show that these are difficult problems to solve, even for deep learning models. Our findings do not provide new theoretical insight into these problems.

Better it went results-wise in the task of separating disciminants based on the number of prime factors in the class numbers. However, discussion 5.2 reveals that one can reach similar classification accuracy through a simple if-then decision rule based on the feature $d \mod 8$.

6.2 Limitations

The experiments conducted in this report were run on a conventional laptop (a MacBook Pro with M1 chip). Naturally, this entails limitations on the scale

and complexity of the experiments undertaken.

In regards to data availability, we limited the scope of the $(h, \mathcal{F}(h))$-dataset by only including h-values in the range $[1, 10^6]$. The reason for this is that the production of $\mathcal{F}(h)$ for large h involves time-consuming computations and since $5 \cdot 10^5$ data points are readily available in [20], we chose to settle for these. Moreover, the `CyPari2` package allows for fast computations of class numbers $h(d)$ through the function `quadclassunit`. Although, these computations slow down as $|d|$ grows. The `quadclassunit` implements McCurley's sub-exponential algorithm for computing the class number $h(d)$ of negative discriminant d [5]. This algorithm has an expected asymptotic average running time of $\mathcal{O}(L(|d|)^{\sqrt{2}})$, where

$$L(|d|) = e^{\sqrt{\log(|d|)\log\log(|d|)}}$$

(see [4, p. 256-259] and [24]).

6.3 Future Work

In regards to predicting the quantity $\mathcal{F}(h)$ based on h, it would be interesting to repeat the training on the feature space $\mathcal{D}_{\mathcal{T}_B^h}$, but with the feature $\mathfrak{c}(h)$ changed to $c_{3,0}(h)$. This would give us performance results of models which do not know $\mathfrak{c}(h)$, but have explicit knowledge regarding whether h is divisible by 3 or not. Likely, this would not yield better results than those we saw in tables 4.1 and 4.2. However, this would still be interesting from an efficiency point of view since computing $\mathfrak{c}(h)$ requires factorisation of h whereas computing $c_{3,0}(h)$ does not. Thus if we could reach comparable results by including $c_{3,0}(h)$ instead of $\mathfrak{c}(h)$ we would have the same performance, but with less computational expense.

The question whether ML models can accurately reveal 3-divisibility of $h(d)$ based on d remains unanswered. Our work, however, shows that a basic setup of an MLP with meagre discriminant feature spaces fails in this task. Further research on this problem is needed to provide a satisfactory answer to the question. The same applies in the case of 5-divisibility.

6.4 Reflections

In this report we have considered tasks of the form: take an integer n as input and predict label ℓ. Bettering established mathematical predictions in

problems of this type — that are known to be difficult — through machine learning algorithms or deep learning, may seem hopeless at first. Indeed, thinking that say a neural network should be able to extract useful and unknown patterns merely from the size of n is pretty naive. However, this seemingly barren area in the landscape of research reveals greater potential than meets the eye. To illustrate the potential of this type of research, suppose that the class number formula 1.2.1 had not yet been discovered. In this scenario, our findings on the probability that $d \equiv 1 \mod 8$ given a certain value of $\Omega(h(d))$ would imply that $d \mod 8$ (which is either 1 or 5) has a significant impact on the size of $h(d)$. Thus, this discovery would have provided a deeper understanding of class numbers.

When it comes to feature engineering and extracting attributes from mathematical objects, it is important to be aware of the cost of these decisions. The awareness necessity of course depends on the research objective. If the primary goal is to analyze feature importance scores purely for the sake of exploration, the cost of constructing feature vectors may be of lesser concern. Conversely, if time efficiency is paramount, careful deliberation is warranted before introducing additional features. For instance, the time complexity of converting an integer n to its binary form is typically $\mathcal{O}(\log(n))$, whereas factoring the integer n entails considerably higher computational expense.

References

[1] T. Trinh, Y. T. Wu, Q. Le, H. He, and T. Luong, "Solving olympiad geometry without human demonstrations," *Nature*, vol. 625, pp. 476–482, 2024. [Online]. Available: https://www.nature.com/articles/s41586-023-06747-5 [Page 1.]

[2] M. Haenlein and A. M. Kaplan, "A brief history of artificial intelligence: On the past, present, and future of artificial intelligence," *California Management Review*, vol. 61, pp. 14 – 5, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:199866730 [Page 1.]

[3] A. Davies, P. Velickovic, L. Buesing, S. Blackwell, D. Zheng, N. Tomasev, R. Tanburn, P. W. Battaglia, C. Blundell, A. Juhasz, M. Lackenby, G. Williamson, D. Hassabis, and P. Kohli, "Advancing mathematics by guiding human intuition with ai," *Nature*, vol. 600, pp. 70 – 74, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:244837059 [Page 1.]

[4] H. Cohen, *A Course in Computational Algebraic Number Theory*. Springer, 1993. [Pages 3, 6, 9, and 54.]

[5] *PARI/GP version 2.11.1*, The PARI Group, Bordeaux, 2018, available from http://pari.math.u-bordeaux.fr/. [Pages 5, 22, and 54.]

[6] D. Goldfeld, "Gauss' class number problem for imaginary quadratic fields," *Bulletin of the American Mathematical Society*, vol. 13, no. 1, p. 23–37, 1985. [Page 5.]

[7] M. Watkins, "Class numbers of imaginary quadratic fields," *Mathematics of Computation*, vol. 73, no. 246, p. 907–938, Oct 2003. doi: 10.1090/s0025-5718-03-01517-5 [Pages 7 and 50.]

[8] D. A. Cox, *Primes of the form $x^2 + ny^2$: Fermat, class field theory, and complex multiplication*. Wiley, 1989. [Page 7.]

[9] S. Holmin, N. Jones, P. Kurlberg, C. McLeman, and K. Petersen, "Missing class groups and class number statistics for imaginary quadratic fields," *Experimental Mathematics*, vol. 28, no. 2, p. 233–254, 11 2017. doi: 10.1080/10586458.2017.1383952 [Pages 7, 10, and 11.]

[10] K. Soundararajan, "The number of imaginary quadratic fields with a given class number," 2007. [Page 9.]

[11] A. Stekel, M. Shukrun, and A. Azaria, "Goldbach's function approximation using deep learning," *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, 2018. doi: 10.1109/WI.2018.00-46 [Page 11.]

[12] D. Wu, J. Yang, M. U. Ahsan, and K. Wang, "Classification of integers based on residue classes via modern deep learning algorithms," *Patterns*, vol. 4, no. 12, p. 100860, Dec. 2023. doi: 10.1016/j.patter.2023.100860. [Online]. Available: http://dx.doi.org/10.1016/j.patter.2023.100860 [Page 12.]

[13] J. Heaton, "An empirical analysis of feature engineering for predictive modeling," in *SoutheastCon 2016*. IEEE, Mar. 2016. doi: 10.1109/secon.2016.7506650. [Online]. Available: http://dx.doi.org/10.1109/SECON.2016.7506650 [Page 12.]

[14] Y.-H. He, "Machine-learning mathematical structures," 2021. [Pages 12 and 13.]

[15] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020. [Pages 13 and 15.]

[16] "The coq proof assistant." [Online]. Available: https://coq.inria.fr/ [Page 13.]

[17] "Lean." [Online]. Available: https://lean-lang.org/ [Page 13.]

[18] G. Gonthier, "Formal proof—the four- color theorem," 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:12620754 [Page 13.]

[19] S. Polu and I. Sutskever, "Generative language modeling for automated theorem proving," *arXiv preprint arXiv:2009.03393*, 2020. [Page 13.]

[20] S. Holmin and P. Kurlberg, "List of $\mathcal{F}(h)$ for all odd $h < 10^6$," https://people.kth.se/~kurlberg/class_group_data/class_group_orders.txt. [Pages 16 and 54.]

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. [Page 20.]

[22] G. Strang, *Linear Algebra and Learning from Data*. Wellesley Cambridge Press, 2019. [Page 24.]

[23] K. Xu, M. Zhang, J. Li, S. S. Du, K.-i. Kawarabayashi, and S. Jegelka, "How neural networks extrapolate: From feedforward to graph neural networks," *arXiv preprint arXiv:2009.11848*, 2020. [Page 48.]

[24] J. L. Hafner and K. S. McCurley, "A rigorous subexponential algorithm for computation of class groups," *Journal of the American Mathematical Society*, vol. 2, pp. 837–850, 1989. [Online]. Available: https://api.semanticscholar.org/CorpusID:51781105 [Page 54.]

[25] R. Crandall and C. Pomerance, *Prime Numbers - A Computational Perspective*. Springer, 2005. [Page 62.]

Appendix A

The Form Class Group

In this appendix we define the class group of binary quadratic forms.

Definition A.0.1. A *binary quadratic form* is a function in two variables of the form $f(x, y) = ax^2 + bxy + cy^2$, where $a, b, c \in \mathbb{Z}$. The *discriminant* of f is defined as $\Delta := b^2 - 4ac$. For convience, we will sometimes denote f simply by (a, b, c).

We only consider negative discriminants and forms (a, b, c) with $a > 0$ and $c > 0$.

Definition A.0.2. Two binary quadratic forms f and g are said to be *equivalent*, if there exists

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \in \mathrm{SL}_2(\mathbb{Z})$$

such that $g(x, y) = f(\alpha x + \beta y, \gamma x + \delta y)$. If f and g are equivalent, we write $f \sim g$.

Importantly, two equivalent binary quadratic forms have the same discriminant (the converse is not true in general).

Definition A.0.3. We say that a binary quadratic form (a, b, c) is *primitive* if $\gcd(a, b, c) = 1$.

Let Δ be a square-free integer such that $\Delta \equiv 0, 1 \mod 4$ and let $\mathcal{C}(\Delta)$ be the set of all equivalence classes of primitive binary quadratic forms of discriminant Δ. The aim is to make $\mathcal{C}(\Delta)$ into a group. For this, we need the following result.

Theorem A.0.1. Let (a, b, c) and (d, e, f) be two primitive binary quadratic forms, both of discriminant Δ. Then there exists binary quadratic forms $(A, \mathscr{C}, C)$ and $(D, \mathscr{C}, F)$ such that $(a, b, c) \sim (A, \mathscr{C}, C)$, $(d, e, f) \sim (D, \mathscr{C}, F)$ and $\gcd(A, D) = 1$.

Proof. See [25, p. 245 - 246]. □

For a primitive binary quadratic form (a, b, c), let $\langle a, b, c\rangle$ denote the equivalence class that contains (a, b, c), i.e. $\langle a, b, c\rangle := [(a, b, c)]$.

Consider forms (a, b, c), (d, e, f), $(A, \mathscr{C}, C)$ and $(D, \mathscr{C}, F)$ as in theorem A.0.1 and consider the operation $*$ on $\mathcal{C}(\Delta)$ defined by

$$\langle a, b, c\rangle * \langle d, e, f\rangle = \langle AD, \mathscr{C}, C/D\rangle.$$

This operation makes $\mathcal{C}(\Delta)$ into an abelian group — the class group of primitive binary quadratic forms of discriminant Δ (see [25, p. 239 - 246] for more details).

Appendix B

Computing Coefficients

In this appendix we show how one can compute the coefficients discussed in section 3.1.3 in an efficient manner. By simplifying and rewriting expressions, we show how to compute c_k for $k = 1, 2, 3, 4$ by essentially using only one for-loop over an arbitrary number of primes.

Recall that we set $L_p \coloneqq 1 - \mathbb{Y}(p)/p$, and let

$$\psi_{p,k} \coloneqq \mathbb{E}\left(L_p^2 \log^k(L_p)\right) = \frac{1}{2}\left(\left(1-\frac{1}{p}\right)^2 \log^k\left(1-\frac{1}{p}\right) + \left(1+\frac{1}{p}\right)^2 \log^k\left(1+\frac{1}{p}\right)\right).$$

Computing c_1

$$E_1 = \mathbb{E}\left(\log\left(\prod_p L_p^{-1}\right)\prod_r L_r^2\right) = \mathbb{E}\left(-\sum_p \log(L_p)\prod_r L_r^2\right) = -\sum_p \mathbb{E}\left(\log(L_p)\prod_r L_r^2\right) =$$
$$-\sum_p \mathbb{E}\left(L_p^2\log(L_p)\prod_{r\neq p} L_r^2\right) = -\sum_p \mathbb{E}\left(L_p^2\log(L_p)\right)\cdot\mathbb{E}\left(\prod_{r\neq p} L_r^2\right)$$
$$-\sum_p \mathbb{E}(L_p^2\log(L_p))\prod_{r\neq p}\mathbb{E}(L_r^2) = -E_0\sum_p \frac{\mathbb{E}(L_p^2\log(L_p))}{\mathbb{E}(L_p^2)} = -E_0\sum_p \frac{\psi_{p,1}}{\psi_{p,0}}.$$

Thus, $c_1 = -\sum_p \psi_{p,1}/\psi_{p,0} \approx -0.578072$.

Computing c_2

$$E_2 = \mathbb{E}\left(\log^2\left(\prod_p L_p^{-1}\right)\prod_r L_r^2\right) = \mathbb{E}\left(\sum_p \log(L_p)\sum_q \log(L_q)\prod_r L_r^2\right) =$$

$$\mathbb{E}\left(\sum_p \log(L_p)^2\prod_r L_r^2\right) + \mathbb{E}\left(\sum_{\substack{p,q\\ \text{distinct}}} \log(L_p)\log(L_q)\prod_r L_r^2\right) =$$

$$\mathbb{E}\left(\sum_p L_p^2\log(L_p)^2\prod_{r\neq p} L_r^2\right) + \mathbb{E}\left(\sum_{\substack{p,q\\ \text{distinct}}} L_p^2\log(L_p)L_q^2\log(L_q)\prod_{r\neq p,q} L_r^2\right) =$$

$$E_0\sum_p \frac{\psi_{p,2}}{\psi_{p,0}} + E_0\sum_{\substack{p,q\\ \text{distinct}}} \frac{\psi_{p,1}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}}.$$

By noting that

$$\sum_{\substack{p,q\\ \text{distinct}}} \frac{\psi_{p,1}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}} = \sum_p \frac{\psi_{p,1}}{\psi_{p,0}}\sum_{q\neq p}\frac{\psi_{q,1}}{\psi_{q,0}} = \sum_p \frac{\psi_{p,1}}{\psi_{p,0}}\left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}}\right) = c_1^2 - \sum_p \frac{\psi_{p,1}^2}{\psi_{p,0}^2},$$

we finally get

$$c_2 = c_1^2 + \sum_p \frac{\psi_{p,2}\psi_{p,0} - \psi_{p,1}^2}{\psi_{p,0}^2} \approx 0.604050.$$

Computing c_3

$$-E_3 = \mathbb{E}\left(\sum_p \log(L_p)\sum_q \log(L_q)\sum_s \log(L_s)\prod_r L_r^2\right) =$$

$$\mathbb{E}\left(\sum_p L_p^2\log(L_p)^3\prod_{r\neq p} L_r^2\right) + 3\mathbb{E}\left(\sum_{\substack{p,q\\ \text{distinct}}} L_p^2\log(L_p)^2L_q^2\log(L_q)\prod_{r\neq p,q} L_r^2\right) +$$

$$\mathbb{E}\left(\sum_{\substack{p,q,s\\ \text{distinct}}} L_p^2\log(L_p)L_q^2\log(L_q)L_s^2\log(L_s)\prod_{r\neq p,q,s} L_r^2\right) =$$

$$E_0\sum_p \frac{\psi_{p,3}}{\psi_{p,0}} + 3E_0\sum_{\substack{p,q\\ \text{distinct}}} \frac{\psi_{p,2}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}} + E_0\sum_{\substack{p,q,s\\ \text{distinct}}} \frac{\psi_{p,1}\psi_{q,1}\psi_{s,1}}{\psi_{p,0}\psi_{q,0}\psi_{s,0}}.$$

Moreover, we have

$$\sum_{\substack{p,q \\ \text{distinct}}} \frac{\psi_{p,2}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}} = \sum_p \frac{\psi_{p,2}}{\psi_{p,0}} \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{p,0}} = \sum_p \frac{\psi_{p,2}}{\psi_{p,0}} \left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}} \right),$$

and

$$\sum_{\substack{p,q,s \\ \text{distinct}}} \frac{\psi_{p,1}\psi_{q,1}\psi_{s,1}}{\psi_{p,0}\psi_{q,0}\psi_{s,0}} = \sum_p \frac{\psi_{p,1}}{\psi_{p,0}} \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \sum_{s \neq p,q} \frac{\psi_{s,1}}{\psi_{s,0}} =$$

$$\sum_p \frac{\psi_{p,1}}{\psi_{p,0}} \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}} - \frac{\psi_{q,1}}{\psi_{q,0}} \right) =$$

$$\sum_p \frac{\psi_{p,1}}{\psi_{p,0}} \left(\left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}} \right)^2 - \sum_{q \neq p} \frac{\psi_{q,1}^2}{\psi_{q,0}^2} \right) =$$

$$\sum_p \frac{\psi_{p,1}}{\psi_{p,0}} \left(c_1^2 + 2c_1 \frac{\psi_{p,1}}{\psi_{p,0}} + \frac{\psi_{p,1}^2}{\psi_{p,0}^2} + \left(c_2 - c_1^2 + \frac{\psi_{p,1}^2}{\psi_{p,0}^2} - \sum_q \frac{\psi_{q,2}}{\psi_{q,0}} \right) \right).$$

Using these results in combination with the fact that

$$\sum_p \frac{\psi_{p,1}}{\psi_{p,0}} \sum_q \frac{\psi_{q,2}}{\psi_{q,0}} = \sum_p \frac{\psi_{p,2}}{\psi_{p,0}} \sum_q \frac{\psi_{q,1}}{\psi_{q,0}} = \sum_p \frac{\psi_{p,2}\psi_{p,1}}{\psi_{p,0}^2} + \sum_{\substack{p,q \\ \text{distinct}}} \frac{\psi_{p,2}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}}$$

we arrive at

$$\begin{aligned} c_3 &= -\sum_p \frac{\psi_{p,3}}{\psi_{p,0}} + c_2 \frac{\psi_{p,1}}{\psi_{p,0}} + 2c_1 \frac{\psi_{p,1}^2}{\psi_{p,0}^2} + 2\frac{\psi_{p,1}^3}{\psi_{p,0}^3} - \frac{\psi_{p,2}\psi_{p,1}}{\psi_{p,0}^2} - 2c_1 \frac{\psi_{p,2}}{\psi_{p,0}} - 2\frac{\psi_{p,2}\psi_{p,1}}{\psi_{p,0}^2} \\ &= -\sum_p \frac{\psi_{p,3}}{\psi_{p,0}} + c_2 \frac{\psi_{p,1}}{\psi_{p,0}} + 2c_1 \frac{\psi_{p,1}^2}{\psi_{p,0}^2} + 2\frac{\psi_{p,1}^3}{\psi_{p,0}^3} - 3\frac{\psi_{p,2}\psi_{p,1}}{\psi_{p,0}^2} - 2c_1 \frac{\psi_{p,2}}{\psi_{p,0}} \approx -0.526259. \end{aligned}$$

Computing c_4

$$E_4 = \mathbb{E}\left(\sum_p \log(L_p) \sum_q \log(L_q) \sum_s \log(L_s) \sum_t \log(L_t) \prod_r L_r^2\right) =$$

$$\mathbb{E}\left(\sum_p L_p^2 \log(L_p)^4 \prod_{r\neq p} L_r^2\right) + \frac{1}{2}\binom{4}{2}\mathbb{E}\left(\sum_{\substack{p,q\\ \text{distinct}}} L_p^2 \log(L_p)^2 L_q^2 \log(L_q)^2 \prod_{r\neq p,q} L_r^2\right) +$$

$$\binom{4}{3}\mathbb{E}\left(\sum_{\substack{p,q\\ \text{distinct}}} L_p^2 \log(L_p)^3 L_q^2 \log(L_q) \prod_{r\neq p,q} L_r^2\right) +$$

$$\binom{4}{2}\mathbb{E}\left(\sum_{\substack{p,q,s\\ \text{distinct}}} L_p^2 \log(L_p)^2 L_q^2 \log(L_q) L_s^2 \log(L_s) \prod_{r\neq p,q,s} L_r^2\right) +$$

$$\mathbb{E}\left(\sum_{\substack{p,q,s,t\\ \text{distinct}}} L_p^2 \log(L_p) L_q^2 \log(L_q) L_s^2 \log(L_s) L_t^2 \log(L_t) \prod_{r\neq p,q,s,t} L_r^2\right) =$$

$$E_0 \sum_p \frac{\psi_{p,4}}{\psi_{p,0}} + 3E_0 \sum_{\substack{p,q\\ \text{distinct}}} \frac{\psi_{p,2}\psi_{q,2}}{\psi_{p,0}\psi_{q,0}} + 4E_0 \sum_{\substack{p,q\\ \text{distinct}}} \frac{\psi_{p,3}\psi_{q,1}}{\psi_{p,0}\psi_{q,0}} +$$

$$6E_0 \sum_{\substack{p,q,s\\ \text{distinct}}} \frac{\psi_{p,2}\psi_{q,1}\psi_{s,1}}{\psi_{p,0}\psi_{q,0}\psi_{s,0}} + E_0 \sum_{\substack{p,q,s,t\\ \text{distinct}}} \frac{\psi_{p,1}\psi_{q,1}\psi_{s,1}\psi_{t,1}}{\psi_{p,0}\psi_{q,0}\psi_{s,0}\psi_{t,0}}.$$

By introducing the notations

$$c_1(p) \coloneqq -\sum_{q\neq p} \frac{\psi_{q,1}}{\psi_{q,0}}$$

$$c_2(p) \coloneqq \sum_{q\neq p} \frac{\psi_{q,2}}{\psi_{q,0}} + \sum_{q\neq p} \sum_{s\neq p,q} \frac{\psi_{q,1}}{\psi_{q,0}} \frac{\psi_{s,1}}{\psi_{s,0}}$$

$$c_3(p) \coloneqq -\sum_{q\neq p} \frac{\psi_{q,3}}{\psi_{q,0}} - 3\sum_{q\neq p} \sum_{s\neq p,q} \frac{\psi_{q,2}}{\psi_{q,0}} \frac{\psi_{s,1}}{\psi_{s,0}} - \sum_{q\neq p} \sum_{s\neq p,q} \sum_{t\neq p,q,s} \frac{\psi_{q,1}}{\psi_{q,0}} \frac{\psi_{s,1}}{\psi_{s,0}} \frac{\psi_{t,1}}{\psi_{t,0}}$$

we realize that

$$c_4 = \sum_p \frac{\psi_{p,4}}{\psi_{p,0}} - 3\sum_p \frac{\psi_{p,3}}{\psi_{p,0}} c_1(p) + 3\sum_p \frac{\psi_{p,2}}{\psi_{p,0}} c_2(p) - \sum_p \frac{\psi_{p,1}}{\psi_{p,0}} c_3(p).$$

It remains to express $c_k(p)$ in terms of p for $k = 1, 2, 3$. We have

$$c_1(p) = c_1 + \frac{\psi_{p,1}}{\psi_{p,0}}$$

and

$$\begin{aligned} c_2(p) &= \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}} - \frac{\psi_{q,1}}{\psi_{q,0}} \right) + \sum_{q \neq p} \frac{\psi_{q,2}}{\psi_{q,0}} \\ &= \left(-c_1 - \frac{\psi_{p,1}}{\psi_{p,0}} \right)^2 - \sum_{q \neq p} \frac{\psi_{q,1}^2}{\psi_{q,0}^2} + \sum_{q \neq p} \frac{\psi_{q,2}}{\psi_{q,0}} \\ &= \left(c_1 + \frac{\psi_{p,1}}{\psi_{p,0}} \right)^2 + c_2 - c_1^2 - \frac{\psi_{p,2}}{\psi_{p,0}} + \frac{\psi_{p,1}^2}{\psi_{p,0}^2}. \end{aligned}$$

In regards to $c_3(p)$, we note that

$$-3 \sum_{q \neq p} \frac{\psi_{q,2}}{\psi_{q,0}} \sum_{s \neq p,q} \frac{\psi_{s,1}}{\psi_{s,0}} = 3 \left(c_1 + \frac{\psi_{p,1}}{\psi_{p,0}} \right) \sum_{q \neq p} \frac{\psi_{q,2}}{\psi_{q,0}} + 3 \sum_{q \neq p} \frac{\psi_{q,2}}{\psi_{q,0}} \frac{\psi_{q,1}}{\psi_{q,0}}$$

and

$$\begin{gathered} -\sum_{q \neq p} \sum_{s \neq p,q} \sum_{t \neq p,q,s} \frac{\psi_{q,1}}{\psi_{q,0}} \frac{\psi_{s,1}}{\psi_{s,0}} \frac{\psi_{t,1}}{\psi_{t,0}} = \\ \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \sum_{s \neq p,q} \frac{\psi_{s,1}}{\psi_{s,0}} \left(c_1 + \frac{\psi_{s,1}}{\psi_{s,0}} + \frac{\psi_{q,1}}{\psi_{q,0}} + \frac{\psi_{p,1}}{\psi_{p,0}} \right) = \\ \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \left[- \left(c_1 + \frac{\psi_{q,1}}{\psi_{q,0}} + \frac{\psi_{p,1}}{\psi_{p,0}} \right)^2 + \sum_{s \neq p,q} \frac{\psi_{s,1}^2}{\psi_{s,0}^2} \right] = \\ \left(c_1 + \frac{\psi_{p,1}}{\psi_{p,0}} \right)^2 \left(c_1 + \frac{\psi_{p,1}}{\psi_{p,0}} \right) - 2 \left(c_1 + \frac{\psi_{p,1}}{\psi_{p,0}} \right) \sum_{q \neq p} \frac{\psi_{q,1}^2}{\psi_{q,0}^2} - \\ \sum_{q \neq p} \frac{\psi_{q,1}^3}{\psi_{q,0}^3} + \sum_{q \neq p} \frac{\psi_{q,1}}{\psi_{q,0}} \sum_{s \neq p,q} \frac{\psi_{s,1}^2}{\psi_{s,0}^2}. \end{gathered}$$

The last term in the above equalities can be written as

$$\sum_{q\neq p}\frac{\psi_{q,1}}{\psi_{q,0}}\sum_{s\neq p,q}\frac{\psi_{s,1}^2}{\psi_{s,0}^2}=\sum_{q\neq p}\frac{\psi_{q,1}}{\psi_{q,0}}\left(-c_2-\frac{\psi_{p,1}^2}{\psi_{p,0}^2}-\frac{\psi_{q,1}^2}{\psi_{q,0}^2}+c_1^2+\sum_r\frac{\psi_{r,2}}{\psi_{r,0}}\right)=$$
$$-\left(c_1^2-c_2-\frac{\psi_{p,1}^2}{\psi_{p,0}^2}\right)\left(c_1+\frac{\psi_{p,1}}{\psi_{p,0}}\right)-\sum_{q\neq p}\frac{\psi_{q,1}^3}{\psi_{q,0}^3}-\left(c_1+\frac{\psi_{p,1}}{\psi_{p,0}}\right)\sum_q\frac{\psi_{q,2}}{\psi_{q,0}}.$$

Furthermore we have that

$$3\frac{\psi_{p,1}}{\psi_{p,0}}\sum_{q\neq p}\frac{\psi_{q,2}}{\psi_{q,0}}-\frac{\psi_{p,1}}{\psi_{p,0}}\sum_q\frac{\psi_{q,2}}{\psi_{q,0}}=2\frac{\psi_{p,1}}{\psi_{p,0}}\sum_{q\neq p}\frac{\psi_{q,2}}{\psi_{q,0}}-\frac{\psi_{p,1}}{\psi_{p,0}}\frac{\psi_{p,2}}{\psi_{p,0}}$$

and

$$2\frac{\psi_{p,1}}{\psi_{p,0}}\left(\sum_{q\neq p}\frac{\psi_{q,2}}{\psi_{q,0}}-\sum_{q\neq p}\frac{\psi_{q,1}^2}{\psi_{q,0}^2}\right)=2\frac{\psi_{p,1}}{\psi_{p,0}}\left(c_2-c_1^2+\frac{\psi_{p,1}^2}{\psi_{p,0}^2}-\frac{\psi_{p,2}}{\psi_{p,0}}\right).$$

Now it follows that

$$\begin{aligned}c_3(p)=c_3-c_2c_1+2c_1\frac{\psi_{p,1}^2}{\psi_{p,0}^2}+2\frac{\psi_{p,1}^3}{\psi_{p,0}^3}-3\frac{\psi_{p,2}}{\psi_{p,0}}\frac{\psi_{p,1}}{\psi_{p,0}}-2c_1\frac{\psi_{p,2}}{\psi_{p,0}}+\\ 2\frac{\psi_{p,1}}{\psi_{p,0}}\left(c_2-c_1^2+\frac{\psi_{p,1}^2}{\psi_{p,0}^2}-\frac{\psi_{p,2}}{\psi_{p,0}}\right)-\\ \left(c_1^2-c_2-\frac{\psi_{p,1}^2}{\psi_{p,0}^2}\right)\left(c_1+\frac{\psi_{p,1}}{\psi_{p,0}}\right)+\\ \left(c_1+\frac{\psi_{p,1}}{\psi_{p,0}}\right)^3-\left(c_1+\frac{\psi_{p,1}}{\psi_{p,0}}\right)\frac{\psi_{p,2}}{\psi_{p,0}}.\end{aligned}$$

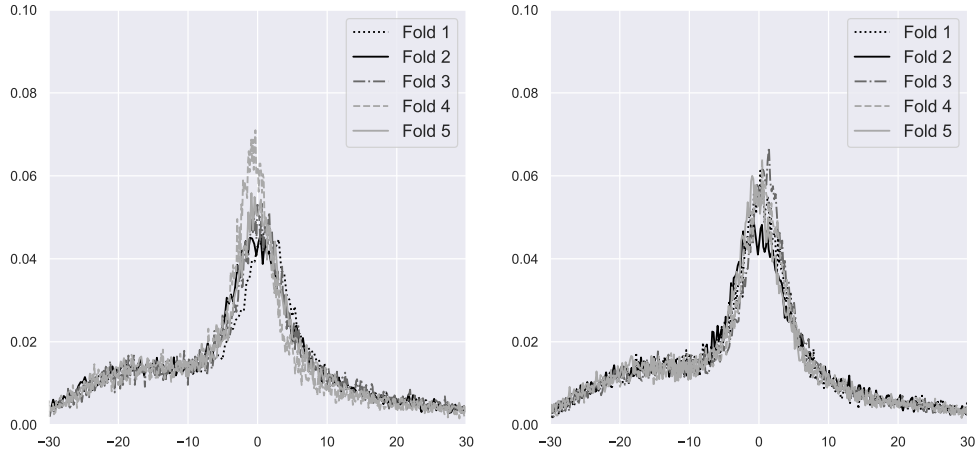
Implementing the above in code yields

$$c_4\approx 0.618741.$$

Appendix C

Supplementary Figures

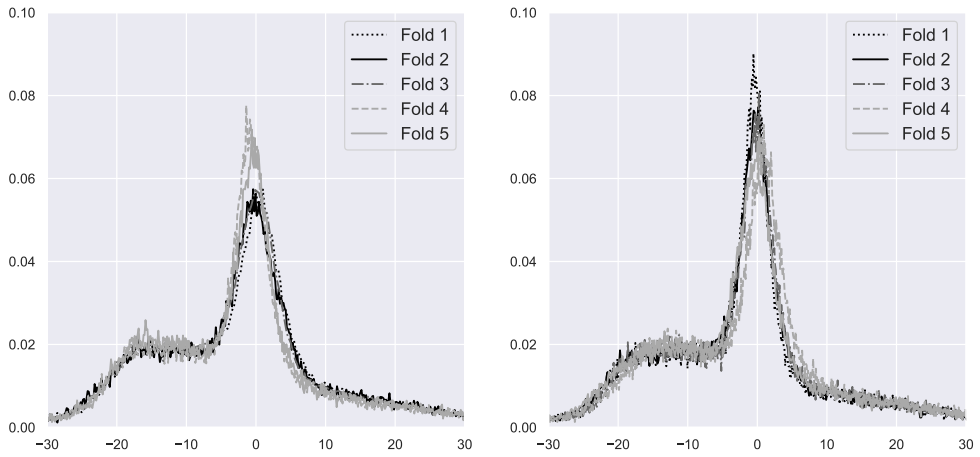
This appendix presents supplementary figures.



(a) Error histograms for $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$. We observed $\mu = 0.26 \pm 0.62$, $\sigma = 21.63 \pm 0.72$.

(b) Error histograms for $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$. We observed $\mu = 0.30 \pm 0.41$, $\sigma = 20.76 \pm 0.15$.

Figure C.1: (Interpolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ for all folds $k = 1, \ldots, 5$. We only consider h such that $3 \mid h$.



(a) Error histograms for $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$. We observed $\mu = -0.32 \pm 0.34$, $\sigma = 16.88 \pm 0.45$.

(b) Error histograms for $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$. We observed $\mu = -0.06 \pm 0.45$, $\sigma = 16.30 \pm 0.18$.

Figure C.2: (Interpolation experiment.) Histograms of scaled test errors $r_{\text{model}}(h)$ for the models $[\mathscr{M}_{\theta_1(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ and $[\mathscr{M}_{\theta_2(k)}, \mathcal{D}_{\mathcal{T}_A^h}]$ for all folds $k = 1, \ldots, 5$. We only consider h such that $3 \nmid h$.

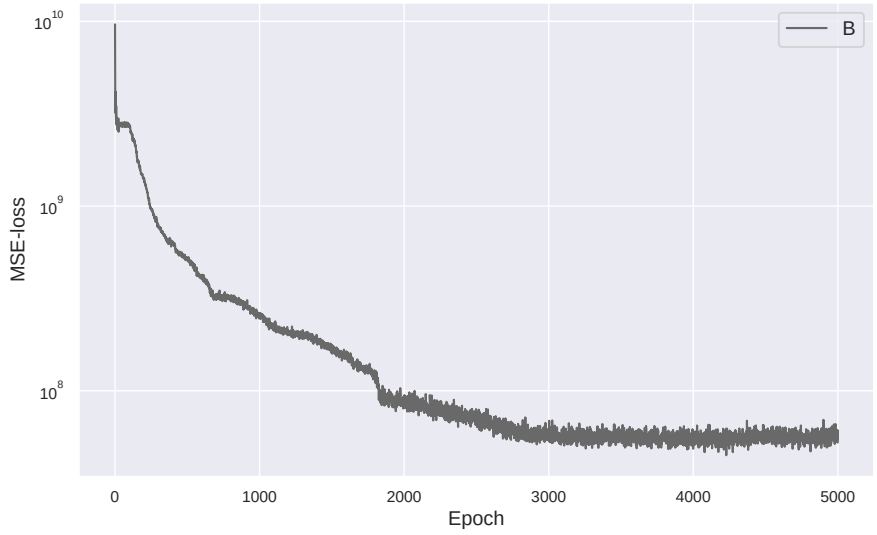
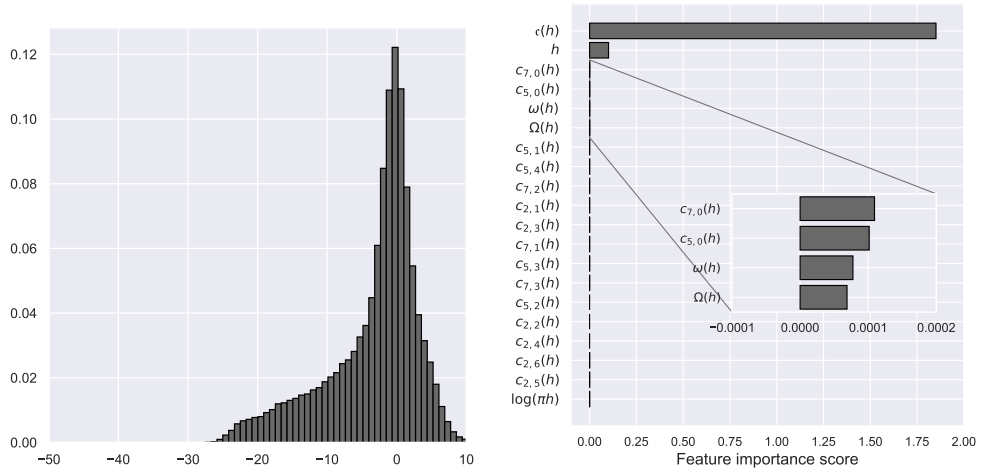


Figure C.3: (Extrapolation experiment.) Test loss for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ over 5000 epochs.



(a) Histogram of scaled test errors $r(h) := (\mathcal{F}(h) - \text{model}(h)/\sqrt{\text{model}(h)})$ for the model $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$.

(b) Feature importance scores in descending order for the model $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$.

Figure C.4: (Extrapolation experiment.) We extended the training for $[\mathscr{M}_{\theta_1}, \mathcal{D}_{\mathcal{T}_B^h}]$ to 5000 epochs and recorded the following evaluation metrics: MSE $\approx 44.785 \cdot 10^6$, $\mu \approx -3.252$ and $\sigma \approx 6.746$. The above figures show the corresponding $r_{\text{model}}(h)$-distribution and feature importance scores.

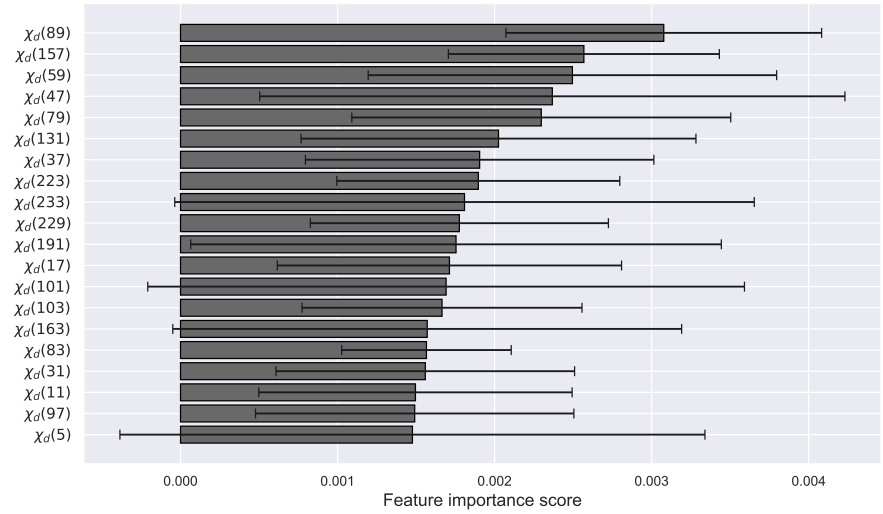


Figure C.5: Top 20 mean feature importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_C^d}]$ on the classification task $3 \mid h(d)$ vs $3 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

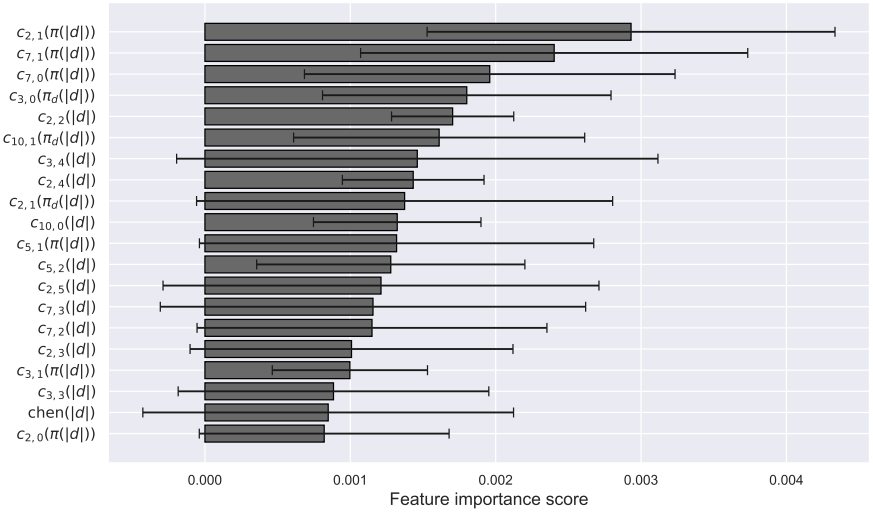


Figure C.6: Top 20 mean feature importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_D^d}]$ on the classification task $5 \mid h(d)$ vs $5 \nmid h(d)$. The error bars indicate one standard deviation over five folds of cross-validation.

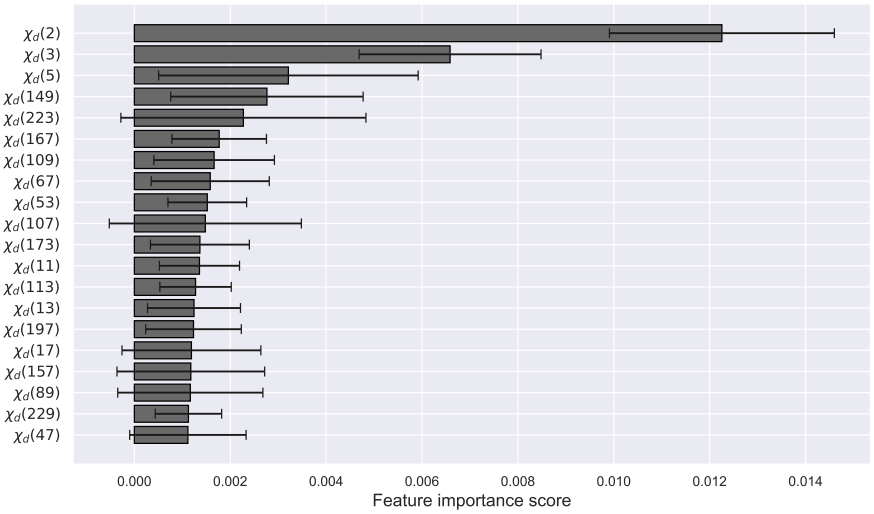


Figure C.7: Top 20 mean feature importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_C^d}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 2$. The error bars indicate one standard deviation over five folds of cross-validation.

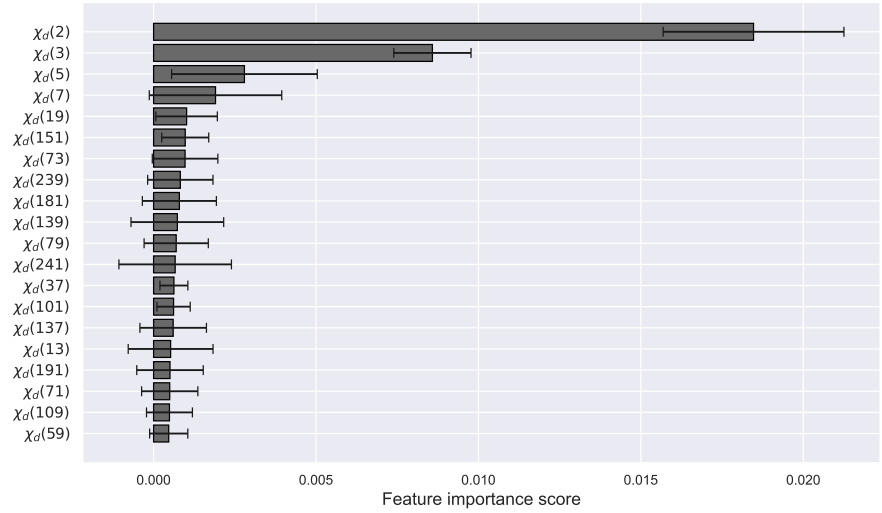


Figure C.8: Top 20 mean feature importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_C^d}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 3$. The error bars indicate one standard deviation over five folds of cross-validation.

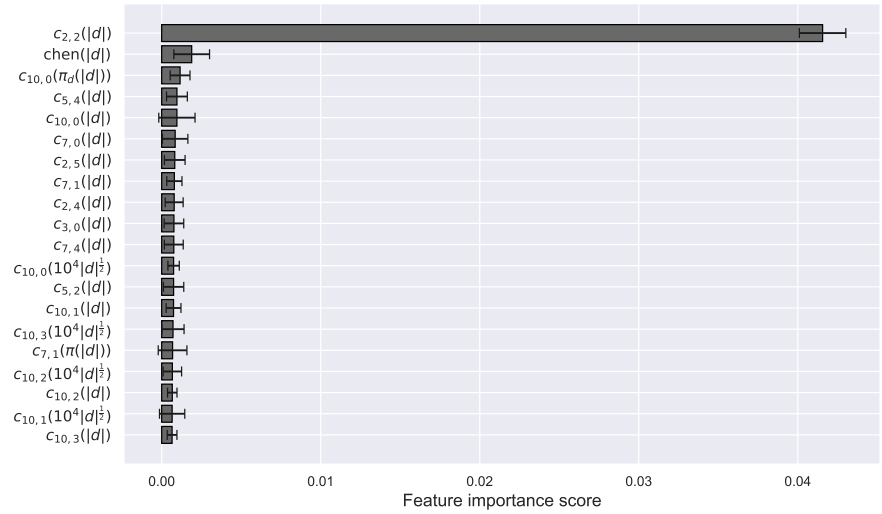


Figure C.9: Top 20 mean feature importance scores for $[\mathscr{M}_\theta, \mathcal{D}_{\mathcal{T}_D^d}]$ on the classification task $\Omega(h(d)) = 1$ vs $\Omega(h(d)) \geq 4$. The error bars indicate one standard deviation over five folds of cross-validation.

TRITA-SCI-EX 2024:0000
Stockholm, Sweden 2024

www.kth.se