# Project 2

Lila Balakrishnan, Andrew Cha, Connor Peace, Oren Romano, Qingran Shao

CSS 383: Bioinformatics

Dr. Zaneveld

## Abstract

As antibiotics have become more easily accessible and have shown to be effective treatments, we have seen a rapid increase in antibiotic resistance. Antibiotic resistance to the quinolone class of antibiotics displays the threat that ARGs pose to the developing medical industry and well being of humans at a global scale. Quinolone antibiotics feature 5 different classes split up into several related subtypes. Of these subtypes, Ciprofloaxcin and Moxifloxacin are two commonly used antibiotics that are categorized under fluoroquinolones, which inhibit the enzymes DNA Gyrase and Topoisomerase IV. In this paper, we look at how different strains of bacteria respond to varying concentrations of the antibiotics Ciprofloxacin and Moxifloxacin.

## Introduction

Ciprofloxacin and Moxifloxacin are antibiotics from the fluoroquinolone class of antibiotics that are commonly used to treat a large range of bacterial infections found throughout the body. These synthetic antibiotics work to defeat the infections by acting on essential enzymes the bacteria need to survive and grow. Without these enzymes functioning properly, it is likely that the DNA will break and cause an increase in mutations. Two of these essential enzymes include DNA gyrase and DNA topoisomerase IV which are involved in supercoiling and translocation that occurs throughout cell growth and division. Shortly after the discovery of these enzymes, it became evident that gyrase was targeted by quinolones. In a study that compared cell exposure to quinolones, wild-type cells with purified gyrase were not supercoiling as they normally would, while *gyrA* mutants seemed to resist the quinolone presence, thus proving that mutated gyrase (*gyrA*) alleles prevent any chromosome breakage when exposed to quinolone (antibiotic resistance).

Another gene that was observed to be quinolone-resistant was the *parC* gene. This gene is considered a subunit of DNA topoisomerase IV, and it was found that a mutation in the *parC* gene leads to a build-up of strong, interlocked molecules that prevent chromosome separation. Through these observations, it was clear that *parC* and *gyrA* were two gene mutations that correlated with antibiotic resistance to Ciprofloxacin and Moxifloxacin.

It is crucial to be able to identify these gene sequences in order to potentially predict resistant strains and apply this epidemiologically to antibiotic resistance. Over the years, the most common methods that have been used to detect similar gene sequences among antibiotic-resistant bacterial strains are PCR testing or microarrays. We are able to access a majority of this data from online databases that provide gene sequences for all or most of the known microbial species, as well as platforms, such as BLAST, that aid in pinpointing the specific *gyrA* and *parC* gene sequences in resistant bacteria.

Based on these findings, we decided to code a program that could assist in predicting antibiotic resistance in bacteria by identifying differences between *gyrA* and *parC* gene sequences of known resistant strains and various other bacterial gene sequences.

*Our specific question:* Can we predict antibiotic resistance based on the differences in *gyrA* and *parC* gene sequences of resistant strains in the regions affected by the antibiotic?

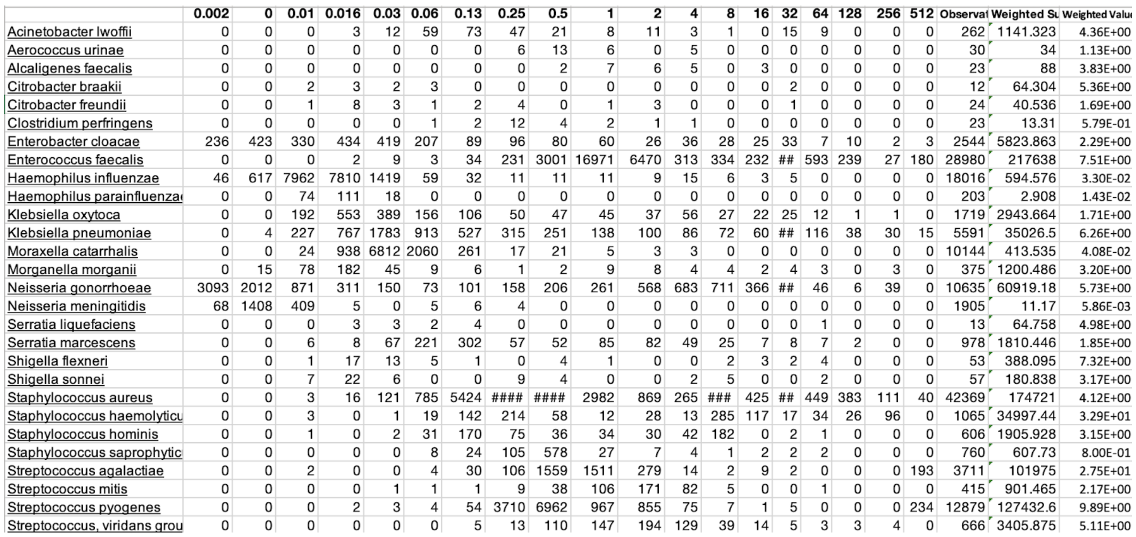
## Materials and Methods

### Measuring resistance

In order to determine antibiotic resistance, we first needed to calculate a weighted number that would serve as a relative rate of resistance to the antibiotic for each strain of bacteria. We were able to do so by looking at MIC data from the EUCAST website. This website provided us with the observed number of surviving bacteria at several different concentrations of Ciprofloxacin and Moxifloxacin; to calculate the relative determinant of antibiotic resistance, we applied this algorithm to each MIC distribution:

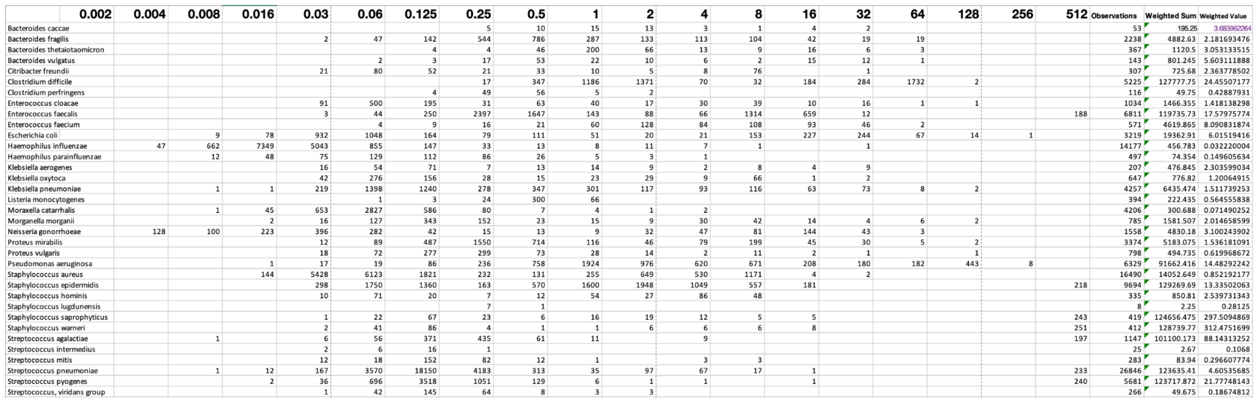
**Σ**(# of surviving microorganisms recorded for the specific concentration * MIC) / the total number of microorganisms observed

**Table 1:** MIC Distributions and Weighted Values for Ciprofloxacin Resistance



| | 0.002 | 0 | 0.01 | 0.016 | 0.03 | 0.06 | 0.13 | 0.25 | 0.5 | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | Observat | Weighted Su | Weighted Valu |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Acinetobacter lwoffii | 0 | 0 | 0 | 3 | 12 | 59 | 73 | 47 | 21 | 8 | 11 | 3 | 1 | 0 | 15 | 9 | 0 | 0 | 0 | 262 | 1141.323 | 4.36E+00 |
| Aerococcus urinae | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 13 | 6 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 30 | 34 | 1.13E+00 |
| Alcaligenes faecalis | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 7 | 6 | 5 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 23 | 88 | 3.83E+00 |
| Citrobacter braakii | 0 | 0 | 2 | 3 | 2 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 12 | 64.304 | 5.36E+00 |
| Citrobacter freundii | 0 | 0 | 1 | 8 | 3 | 1 | 2 | 4 | 0 | 1 | 3 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 24 | 40.536 | 1.69E+00 |
| Clostridium perfringens | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 12 | 4 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 23 | 13.31 | 5.79E-01 |
| Enterobacter cloacae | 236 | 423 | 330 | 434 | 419 | 207 | 89 | 96 | 80 | 60 | 26 | 36 | 28 | 25 | 33 | 7 | 10 | 2 | 3 | 2544 | 5823.863 | 2.29E+00 |
| Enterococcus faecalis | 0 | 0 | 0 | 2 | 9 | 3 | 34 | 231 | 3001 | 16971 | 6470 | 313 | 334 | 232 | ## | 593 | 239 | 27 | 180 | 28980 | 217638 | 7.51E+00 |
| Haemophilus influenzae | 46 | 617 | 7962 | 7810 | 1419 | 59 | 32 | 11 | 11 | 11 | 9 | 15 | 6 | 3 | 5 | 0 | 0 | 0 | 0 | 18016 | 594.576 | 3.30E-02 |
| Haemophilus parainfluenzae | 0 | 0 | 74 | 111 | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 203 | 2.908 | 1.43E-02 |
| Klebsiella oxytoca | 0 | 0 | 192 | 553 | 389 | 156 | 106 | 50 | 47 | 45 | 37 | 56 | 27 | 22 | 25 | 12 | 1 | 1 | 0 | 1719 | 2943.664 | 1.71E+00 |
| Klebsiella pneumoniae | 0 | 4 | 227 | 767 | 1783 | 913 | 527 | 315 | 251 | 138 | 100 | 86 | 72 | 60 | ## | 116 | 38 | 30 | 15 | 5591 | 35026.5 | 6.26E+00 |
| Moraxella catarrhalis | 0 | 0 | 24 | 938 | 6812 | 2060 | 261 | 17 | 21 | 5 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 10144 | 413.535 | 4.08E-02 |
| Morganella morganii | 0 | 15 | 78 | 182 | 45 | 9 | 6 | 1 | 2 | 9 | 8 | 4 | 4 | 2 | 4 | 3 | 0 | 3 | 0 | 375 | 1200.486 | 3.20E+00 |
| Neisseria gonorrhoeae | 3093 | 2012 | 871 | 311 | 150 | 73 | 101 | 158 | 206 | 261 | 568 | 683 | 711 | 366 | ## | 46 | 6 | 39 | 0 | 10635 | 60919.18 | 5.73E+00 |
| Neisseria meningitidis | 68 | 1408 | 409 | 5 | 0 | 5 | 6 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1905 | 11.17 | 5.86E-03 |
| Serratia liquefaciens | 0 | 0 | 0 | 3 | 3 | 2 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 13 | 64.758 | 4.98E+00 |
| Serratia marcescens | 0 | 0 | 6 | 8 | 67 | 221 | 302 | 57 | 52 | 85 | 82 | 49 | 25 | 7 | 8 | 7 | 2 | 0 | 0 | 978 | 1810.446 | 1.85E+00 |
| Shigella flexneri | 0 | 0 | 1 | 17 | 13 | 5 | 1 | 0 | 4 | 1 | 0 | 0 | 2 | 3 | 2 | 4 | 0 | 0 | 0 | 53 | 388.095 | 7.32E+00 |
| Shigella sonnei | 0 | 0 | 7 | 22 | 6 | 0 | 0 | 9 | 4 | 0 | 0 | 2 | 5 | 0 | 0 | 2 | 0 | 0 | 0 | 57 | 180.838 | 3.17E+00 |
| Staphylococcus aureus | 0 | 0 | 3 | 16 | 121 | 785 | 5424 | #### | #### | 2982 | 869 | 265 | ### | 425 | ## | 449 | 383 | 111 | 40 | 42369 | 174721 | 4.12E+00 |
| Staphylococcus haemolyticu | 0 | 0 | 3 | 0 | 1 | 19 | 142 | 214 | 58 | 12 | 28 | 13 | 285 | 117 | 17 | 34 | 26 | 96 | 0 | 1065 | 34997.44 | 3.29E+01 |
| Staphylococcus hominis | 0 | 0 | 1 | 0 | 2 | 31 | 170 | 75 | 36 | 34 | 30 | 42 | 182 | 0 | 2 | 1 | 0 | 0 | 0 | 606 | 1905.928 | 3.15E+00 |
| Staphylococcus saprophytic | 0 | 0 | 0 | 0 | 0 | 8 | 24 | 105 | 578 | 27 | 7 | 4 | 1 | 2 | 2 | 2 | 0 | 0 | 0 | 760 | 607.73 | 8.00E-01 |
| Streptococcus agalactiae | 0 | 0 | 2 | 0 | 0 | 4 | 30 | 106 | 1559 | 1511 | 279 | 14 | 2 | 9 | 2 | 0 | 0 | 0 | 193 | 3711 | 101975 | 2.75E+01 |
| Streptococcus mitis | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 9 | 38 | 106 | 171 | 82 | 5 | 0 | 0 | 1 | 0 | 0 | 0 | 415 | 901.465 | 2.17E+00 |
| Streptococcus pyogenes | 0 | 0 | 0 | 2 | 3 | 4 | 54 | 3710 | 6962 | 967 | 855 | 75 | 7 | 1 | 5 | 0 | 0 | 0 | 234 | 12879 | 127432.6 | 9.89E+00 |
| Streptococcus, viridans grou | 0 | 0 | 0 | 0 | 0 | 0 | 5 | 13 | 110 | 147 | 194 | 129 | 39 | 14 | 5 | 3 | 3 | 4 | 0 | 666 | 3405.875 | 5.11E+00 |

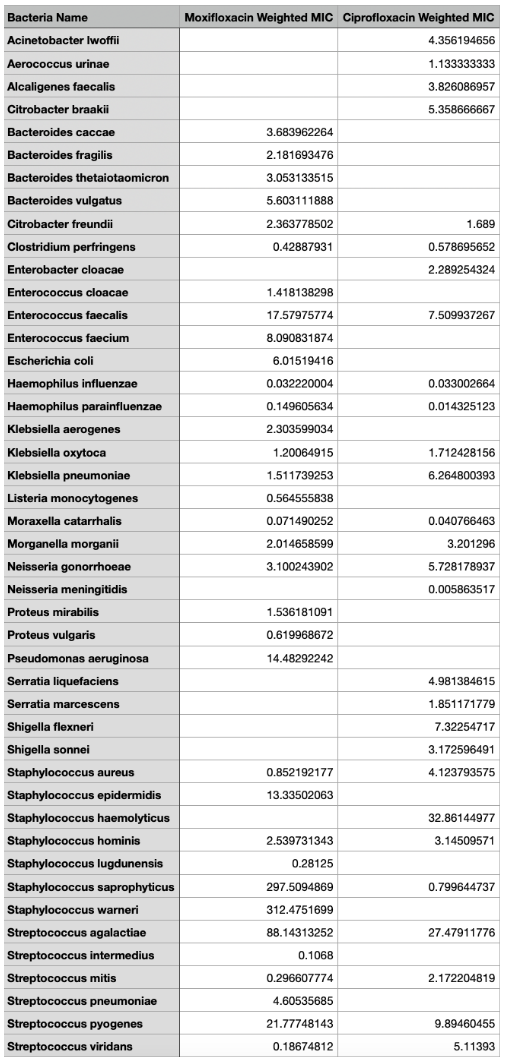
**Table 2:** MIC Distributions and Weighted Values for Moxifloxacin Resistance



| | 0.002 | 0.004 | 0.008 | 0.016 | 0.03 | 0.06 | 0.125 | 0.25 | 0.5 | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | Observations | Weighted Sum | Weighted Value |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bacteroides caccae | | | | | | | | 5 | 10 | 15 | 13 | 3 | 1 | 4 | 2 | | | | | 53 | 195.25 | 3.683962264 |
| Bacteroides fragilis | | | | | 2 | 47 | 142 | 544 | 786 | 287 | 133 | 113 | 104 | 42 | 19 | 19 | | | | 2238 | 4882.63 | 2.181693476 |
| Bacteroides thetaiotaomicron | | | | | | | 4 | 4 | 46 | 200 | 66 | 13 | 9 | 16 | 6 | 3 | | | | 367 | 1120.5 | 3.053133515 |
| Bacteroides vulgatus | | | | | | 2 | 3 | 17 | 53 | 22 | 10 | 6 | 2 | 15 | 12 | 1 | | | | 143 | 801.245 | 5.603111888 |
| Citribacter freundii | | | | | 21 | 80 | 52 | 21 | 33 | 10 | 5 | 8 | 76 | | 1 | | | | | 307 | 725.68 | 2.363778502 |
| Clostridium difficile | | | | | | | | 17 | 347 | 1186 | 1371 | 70 | 32 | 184 | 284 | 1732 | 2 | | | 5225 | 127777.75 | 24.45507177 |
| Clostridium perfringens | | | | | | | 4 | 49 | 56 | 5 | 2 | | | | | | | | | 116 | 49.75 | 0.42887931 |
| Enterococcus cloacae | | | | | 91 | 500 | 195 | 31 | 63 | 40 | 17 | 30 | 39 | 10 | 16 | 1 | 1 | | | 1034 | 1466.355 | 1.418138298 |
| Enterococcus faecalis | | | | | 3 | 44 | 250 | 2397 | 1647 | 143 | 88 | 66 | 1314 | 659 | 12 | | | | 188 | 6811 | 119735.73 | 17.57975774 |
| Enterococcus faecium | | | | | | 4 | 9 | 16 | 21 | 60 | 128 | 84 | 108 | 93 | 46 | 2 | | | | 571 | 4619.865 | 8.090831874 |
| Escherichia coli | | | 9 | 78 | 932 | 1048 | 164 | 79 | 111 | 51 | 20 | 21 | 153 | 227 | 244 | 67 | 14 | 1 | | 3219 | 19362.91 | 6.01519416 |
| Haemophilus influenzae | | 47 | 662 | 7349 | 5043 | 855 | 147 | 33 | 13 | 8 | 11 | 7 | 1 | | 1 | | | | | 14177 | 456.783 | 0.032220004 |
| Haemophilus parainfluenzae | | | 12 | 48 | 75 | 129 | 112 | 86 | 26 | 5 | 3 | 1 | | | | | | | | 497 | 74.354 | 0.149605634 |
| Klebsiella aerogenes | | | | | 16 | 54 | 71 | 7 | 13 | 14 | 9 | 2 | 8 | 4 | 9 | | | | | 207 | 476.845 | 2.303599034 |
| Klebsiella oxytoca | | | | | 42 | 276 | 156 | 28 | 15 | 23 | 29 | 9 | 66 | 1 | 2 | | | | | 647 | 776.82 | 1.20064915 |
| Klebsiella pneumoniae | | | 1 | 1 | 219 | 1398 | 1240 | 278 | 347 | 301 | 117 | 93 | 116 | 63 | 73 | 8 | 2 | | | 4257 | 6435.474 | 1.511739253 |
| Listeria monocytogenes | | | | | | 1 | 3 | 24 | 300 | 66 | | | | | | | | | | 394 | 222.435 | 0.564555838 |
| Moraxella catarrhalis | | | 1 | 45 | 653 | 2827 | 586 | 80 | 7 | 4 | 1 | 2 | | | | | | | | 4206 | 300.688 | 0.071490252 |
| Morganella morganii | | | | 2 | 16 | 127 | 343 | 152 | 23 | 15 | 9 | 30 | 42 | 14 | 4 | 6 | 2 | | | 785 | 1581.507 | 2.014658599 |
| Neisseria gonorrhoeae | | 128 | 100 | 223 | 396 | 282 | 42 | 15 | 13 | 9 | 32 | 47 | 81 | 144 | 43 | 3 | | | | 1558 | 4830.18 | 3.100243902 |
| Proteus mirabilis | | | | | 12 | 89 | 487 | 1550 | 714 | 116 | 46 | 79 | 199 | 45 | 30 | 5 | 2 | | | 3374 | 5183.075 | 1.536181091 |
| Proteus vulgaris | | | | | 18 | 72 | 277 | 299 | 73 | 28 | 14 | 2 | 11 | 2 | 1 | | 1 | | | 798 | 494.735 | 0.619968672 |
| Pseudomonas aeruginosa | | | | 1 | 17 | 19 | 86 | 236 | 758 | 1924 | 976 | 620 | 671 | 208 | 180 | 182 | 443 | 8 | | 6329 | 91662.416 | 14.48292242 |
| Staphylococcus aureus | | | | 144 | 5428 | 6123 | 1821 | 232 | 131 | 255 | 649 | 530 | 1171 | 4 | 2 | | | | | 16490 | 14052.649 | 0.852192177 |
| Staphylococcus epidermidis | | | | | 298 | 1750 | 1360 | 163 | 570 | 1600 | 1948 | 1049 | 557 | 181 | | | | | 218 | 9694 | 129269.69 | 13.33502063 |
| Staphylococcus hominis | | | | | 10 | 71 | 20 | 7 | 12 | 54 | 27 | 86 | 48 | | | | | | | 335 | 850.81 | 2.539731343 |
| Staphylococcus lugdunensis | | | | | | | | 7 | 1 | | | | | | | | | | | 8 | 2.25 | 0.28125 |
| Staphylococcus saprophyticus | | | | | 1 | 22 | 67 | 23 | 6 | 16 | 19 | 12 | 5 | 5 | | | | | 243 | 419 | 124656.475 | 297.5094869 |
| Staphylococcus warneri | | | | | 2 | 41 | 86 | 4 | 1 | 1 | 6 | 6 | 6 | 8 | | | | | 251 | 412 | 128739.77 | 312.4751699 |
| Streptococcus agalactiae | | | 1 | | 6 | 56 | 371 | 435 | 61 | 11 | | 9 | | | | | | | 197 | 1147 | 101100.173 | 88.14313252 |
| Streptococcus intermedius | | | | | 2 | 6 | 16 | 1 | | | | | | | | | | | | 25 | 2.67 | 0.1068 |
| Streptococcus mitis | | | | | 12 | 18 | 152 | 82 | 12 | 1 | | 3 | 3 | | | | | | | 283 | 83.94 | 0.296607774 |
| Streptococcus pneumoniae | | | 1 | 12 | 167 | 3570 | 18150 | 4183 | 313 | 35 | 97 | 67 | 17 | 1 | | | | | 233 | 26846 | 123635.41 | 4.60535685 |
| Streptococcus pyogenes | | | | 2 | 36 | 696 | 3518 | 1051 | 129 | 6 | 1 | 1 | | 1 | | | | | 240 | 5681 | 123717.872 | 21.77748143 |
| Streptococcus, viridans group | | | | | 1 | 42 | 145 | 64 | 8 | 3 | 3 | | | | | | | | | 266 | 49.675 | 0.18674812 |

Using these weighted numbers, we were then able to determine a comparable level of resistance among the strains.

**Table 3:** Condensed table of Weighted MIC Resistance for Moxifloxacin and Ciprofloxacin



| Bacteria Name | Moxifloxacin Weighted MIC | Ciprofloxacin Weighted MIC |
|---|---|---|
| Acinetobacter lwoffii | | 4.356194656 |
| Aerococcus urinae | | 1.133333333 |
| Alcaligenes faecalis | | 3.826086957 |
| Citrobacter braakii | | 5.358666667 |
| Bacteroides caccae | 3.683962264 | |
| Bacteroides fragilis | 2.181693476 | |
| Bacteroides thetaiotaomicron | 3.053133515 | |
| Bacteroides vulgatus | 5.603111888 | |
| Citrobacter freundii | 2.363778502 | 1.689 |
| Clostridium perfringens | 0.42887931 | 0.578695652 |
| Enterobacter cloacae | | 2.289254324 |
| Enterococcus cloacae | 1.418138298 | |
| Enterococcus faecalis | 17.57975774 | 7.509937267 |
| Enterococcus faecium | 8.090831874 | |
| Escherichia coli | 6.01519416 | |
| Haemophilus influenzae | 0.032220004 | 0.033002664 |
| Haemophilus parainfluenzae | 0.149605634 | 0.014325123 |
| Klebsiella aerogenes | 2.303599034 | |
| Klebsiella oxytoca | 1.20064915 | 1.712428156 |
| Klebsiella pneumoniae | 1.511739253 | 6.264800393 |
| Listeria monocytogenes | 0.564555838 | |
| Moraxella catarrhalis | 0.071490252 | 0.040766463 |
| Morganella morganii | 2.014658599 | 3.201296 |
| Neisseria gonorrhoeae | 3.100243902 | 5.728178937 |
| Neisseria meningitidis | | 0.005863517 |
| Proteus mirabilis | 1.536181091 | |
| Proteus vulgaris | 0.619968672 | |
| Pseudomonas aeruginosa | 14.48292242 | |
| Serratia liquefaciens | | 4.981384615 |
| Serratia marcescens | | 1.851171779 |
| Shigella flexneri | | 7.32254717 |
| Shigella sonnei | | 3.172596491 |
| Staphylococcus aureus | 0.852192177 | 4.123793575 |
| Staphylococcus epidermidis | 13.33502063 | |
| Staphylococcus haemolyticus | | 32.86144977 |
| Staphylococcus hominis | 2.539731343 | 3.14509571 |
| Staphylococcus lugdunensis | 0.28125 | |
| Staphylococcus saprophyticus | 297.5094869 | 0.799644737 |
| Staphylococcus warneri | 312.4751699 | |
| Streptococcus agalactiae | 88.14313252 | 27.47911776 |
| Streptococcus intermedius | 0.1068 | |
| Streptococcus mitis | 0.296607774 | 2.172204819 |
| Streptococcus pneumoniae | 4.60535685 | |
| Streptococcus pyogenes | 21.77748143 | 9.89460455 |
| Streptococcus viridans | 0.18674812 | 5.11393 |

**Combining the gyrA and parC FASTA files**

After downloading the gyrA and parC gene sequences as FASTA files from each of the bacteria strains found in **Table 1** and **2,** we needed to combine them into two FASTA files, one for all of the gyrA sequences and one for all of the parC sequences, so that we could perform two multiple sequence alignments. To do this, we wrote a python script that looped through each file in the folder containing each of the sequences and created two separate lists of file names, one for the gyrA sequences and one for the parC sequences. We printed the length of these lists as a manual check to ensure they were the same length. These lists allowed us to use a single *combine_fasta* method for both genes which accepted a list of filenames and an output file name and output a combined FASTA with the given name.

The method to create the combined FASTA was straightforward. First, we opened our output file with the given file name, throwing an exception if a file with that name already existed. Then, we first checked if the file existed that we were trying to read. If any files didn't exist, something that we did not expect to happen with our method of creating the file names, an error would print to the console. Next, we opened the file and read its contents into the combined FASTA file, including its identification line. This repeated for each file name until the end of the list of file names was reached, and we closed and saved the combined FASTA file.

The combined FASTA files were uploaded to our shared Google Drive folder with names that were agreed upon in advance so that our team could perform the multiple sequence alignments.

**Multiple sequence alignment**

As a pairwise sequence alignment will set two sequences into a 2D plane to compare, multiple sequence alignments will create an N-degree matrix to find the best alignment of the N sequences. The challenge is how two control the granularity or accuracy when comparing multiple data. In sequence alignment, it is easy to set the average weight of two pieces of sequences, but it is hard to tell which sequence is a priority when the number of sequences is above 3. To deal with this scenario, MUSCLE provides a good solution. The combined FASTA files created by our group were each inputted into the MUSCLE sequence alignment tool to provide an optimal sequence alignment.

**Sequence processing for modeling**

In order to create machine learning models to predict the MIC resistance levels of bacteria, we needed to develop a way to convert the DNA sequences into numerical equivalents. Initially, we used a percent identity matrix (PIM) comparing the gene sequence of all the species in our dataset to each other. The PIM was created using the ClustalW software. However, the result of this technique was that we ended up with an *nxn* matrix of percent comparisons for n species. Since we had no way to distill this information into the relevant comparison numbers for determining the MIC value of a single species, we determined that a more accurate means of evaluating the sequences needed to be used. We then tested using a Transitive Consistency Score (TCS) for each gene of the different species as a predictive value. We generated these numbers using the aligned sequences from earlier and running them through the TCS software developed by TCoffee (Chang et al.). This provided a single value for each DNA sequence—since we were examining two genes for each species, this resulted in two numbers for each bacteria strain. This provided much more useful feature data to use in our machine learning models. In addition to this, we tested using k-merization and vectorization as a means to turn DNA sequences into a more comprehensive dataset. K-merization (the bag of words model), one hot encoding, and vectorization are tools that feature prominently in natural language processing (NLP) and deep learning. Based on the average sequence length of our data, we decided the best length of our "words" should be 6 letters long (hexamers). We initially examined the concept of one hot encoding to establish numerical equivalences for the hexamers. However, due to a large number of hexamers and the fact that each "word" would require six arrays, we rejected using this idea due to the excessive memory and time that would be required. As such, we decided to use the CountVectorization function in sklearn to implement the word-to-vector encoding and used this for further models.

**Hidden Markov model**

The Hidden Markov model is the most complex and esoteric model among shallow mechanical learning models. The basis of the hidden Markov model is the Bayesian principle and Bayesian changes. The problem that this model hopes to solve is to find a model between positive data and potential data, and predict the best model.

In this project, we hope to protect the antibiotic-resistant in MIC by obtaining new DNA. Therefore, we will get the second type of HMM suitable: machining learning problem by known DNA sequence, through model simulation to obtain better DNA prediction.

The algorithms are the Baum-Welch algorithm and Vibter algorithm. The Baum-Welch algorithm uses the largest application program to obtain the maximum likelihood estimation of hidden Markov model parameters given a set of model features. By finding the frequency of the DNA pair…

$$\{AA,AG,AT,AC\}$$

$$\{GA,GG,GT,GC\}$$

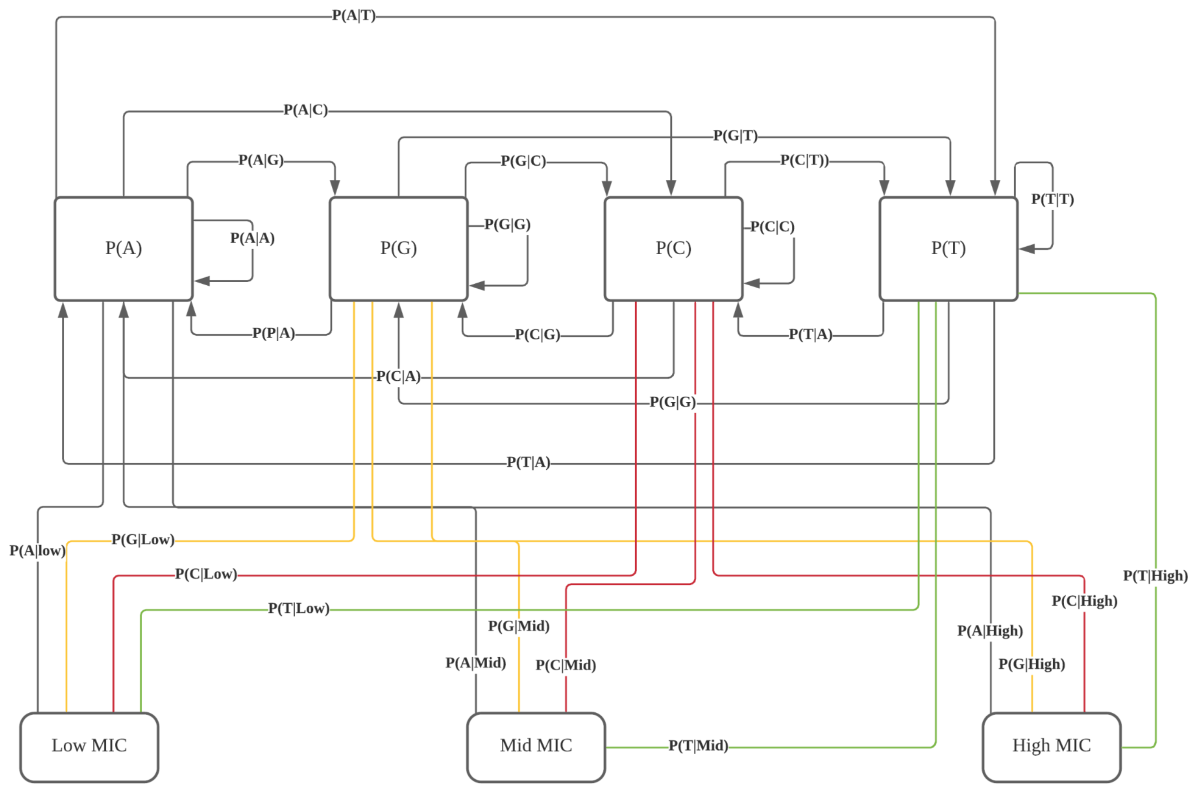
$$\{TA,TG,TT,TC\}$$

$$\{CA,CG,CT,CC\}$$

… the model is able to get a transition matrix about DNA, and then after re-estimating the emission matrix (according to the occurrence of MIC, it can be divided into three states: low, medium, and high). The algorithm will look for the biggest complaint until the complete DNA sequence is read.

The Vibiter algorithm uses this model. By feeding the new DNA sequence to the model and comparing the MIC returned to the actual MIC, the verification of the MIC comes from the synthesis of the model algorithm.

The following picture (HMM-01) shows the HMM in our project:



(HMM-01)

## Discussion

### Issue with overfitting

By using sequences we collected from the PATRIC database for ciprofloxacin and moxifloxacin we expected our accuracy for identifying resistance to increase proportionally with additional sequencing data. Using the MIC weighted values for antibiotic resistance as a comparative point, we established a model aligning similar sequences for each strain of bacteria to isolate possible antibiotic resistance genes. By using multiple sequence alignment for gyrA and parC genes, our model was able to recognize defined antibiotic resistance at a low accuracy with the first set of ciprofloxacin data. The introduction of moxifloxacin data proved to be difficult as some strains of bacteria lacked parC gene features in complete genomes, leading to negative accuracy when run independently. As we combined the data sets together accuracy improved, so negative accuracy may be a result of too little data or inconsistencies between featured data sets.

Another potential feature of our project was the Hidden Markov Model. Due to a lack of time, a complete HMM was unable to come to fruition however it would have proved very useful in establishing a model for antibiotic resistance in bacteria with hidden states. The similarity of potential data is the scale of the data model by establishing a statement model relationship. However, since HMM is a model born based on Bayesian theory, the following assumptions are required to establish an HMM model.

1. The representational data are mutually independent, and the recessive number is (time) homogeneous
2. State: When a random process is in the present and all past states, its future state causes interruption only depends on the current state.

In other words, when the current state is given, it is related to the past state (that is, the historical path of the process). It is conditionally independent.

Due to these two prerequisites, HMM only appears in a small amount of data, and when the amount of data may increase, or it will cause the amount of data to increase simultaneously with HMM. In the specific direction of the DNA sequence, the possibility of each gene is implicitly related to the front and back of the gene, and this may be related to chemical reactions and chemical bonds. Deep learning will perform better than HMM. But it does not mean that HMM loses its meaning at this moment. The currently measured DNA sequence and MIC preparation support deep learning to establish an accurate model. When predicting the antibiotic resistance of new DNA, HMM will get a relatively more accurate estimate. This is the significance of MHH in the project.

Adding additional data and incorporating an HMM would greatly increase the accuracy of our project and allow us to properly identify sequences of antibiotic-resistant genes. An extension of this project would be applying this model to other animal genomes for identifying factors or sequences for diseases. In combination with CRISPR, gene mapping and therapy could reinforce the medical field by eliminating genetically-tied diseases.

## Results

The initial stage of our experiment had very limited data (only 28 bacterial strains of which we could only train the data on 22 points). This led to our prediction accuracy being quite low, averaging at approximately 11% (Supplemental Data Fig. 1). However, our results did seem to show that the ML process could predict the lower resistance levels with higher accuracy (around 64% taking into account the error margin). The low number of data with extremely high resistance levels led to a few outliers that severely impacted our prediction accuracy.

Our low accuracy in our initial experiment led to us expanding our dataset to include other antibiotics in the fluoroquinolone family, specifically moxifloxacin. This added another 33 bacteria species to our dataset. Furthermore, while we expanded our data in terms of the number of strains, we also decided to further develop our method of analyzing the DNA sequences that we were using to predict resistance levels. We implemented the use of a single value similarity score distilled from the TCS. We also decided to test out other machine learning algorithms in order to determine if an alternate option may provide better results. We implemented the Support Vector Regression model, which increased the accuracy of our predictions from the initial state of 11% to 35% accuracy. We implemented the TCS scores as our ML features in the Random Forest Regression model we had been using. By doing so, we were able to increase the accuracy of our prediction model to 63%.

We tested another method of sequence analysis and evaluation in order to determine the impact this had on our models. Using the DNA sequence strings, we divided each into a set of overlapping hexamers ("words" six letters long). Using a word-to-vector technique, we were able to create numerical representations of each sequence, and use that as our features data for the machine learning models. This k-mer sequence representation was tested in our Random Forest Regression model and our Support Vector Regression model. Using this, our models gave us a 76% and 36% accuracy, respectively.

While this was not a significant improvement for the SVR model, there was a significant increase in accuracy using Random Forest. Our experimental model overall increased in accuracy from 11% to 76% over the course of our experiment.

We aimed to test some of these data combinations in a Hidden Markov Model, however, implementing it in our current timeframe was not feasible. In the future, we hope to be able to test out the HMM and further experiment in order to optimize our solution to this problem.

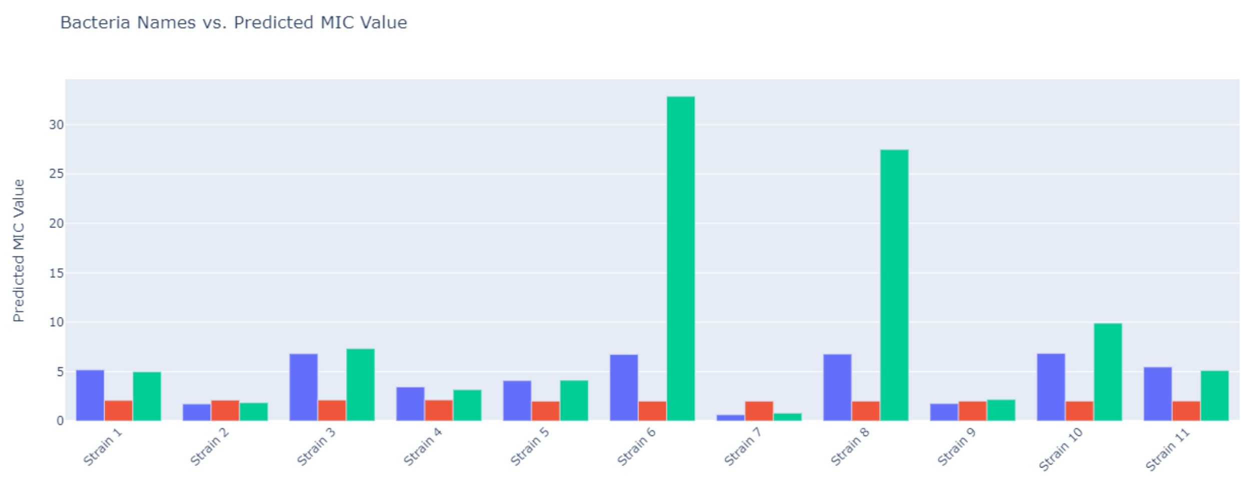
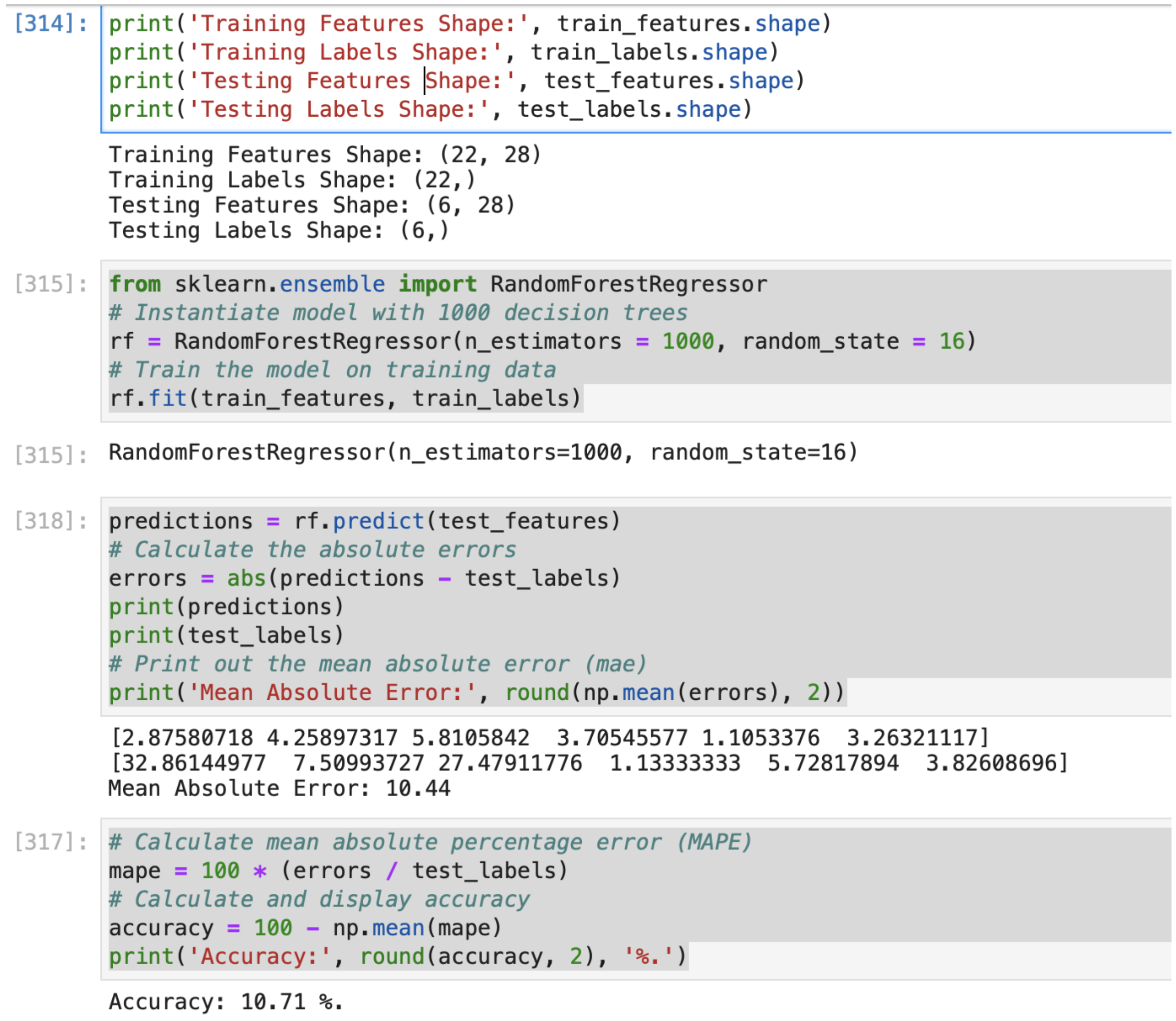
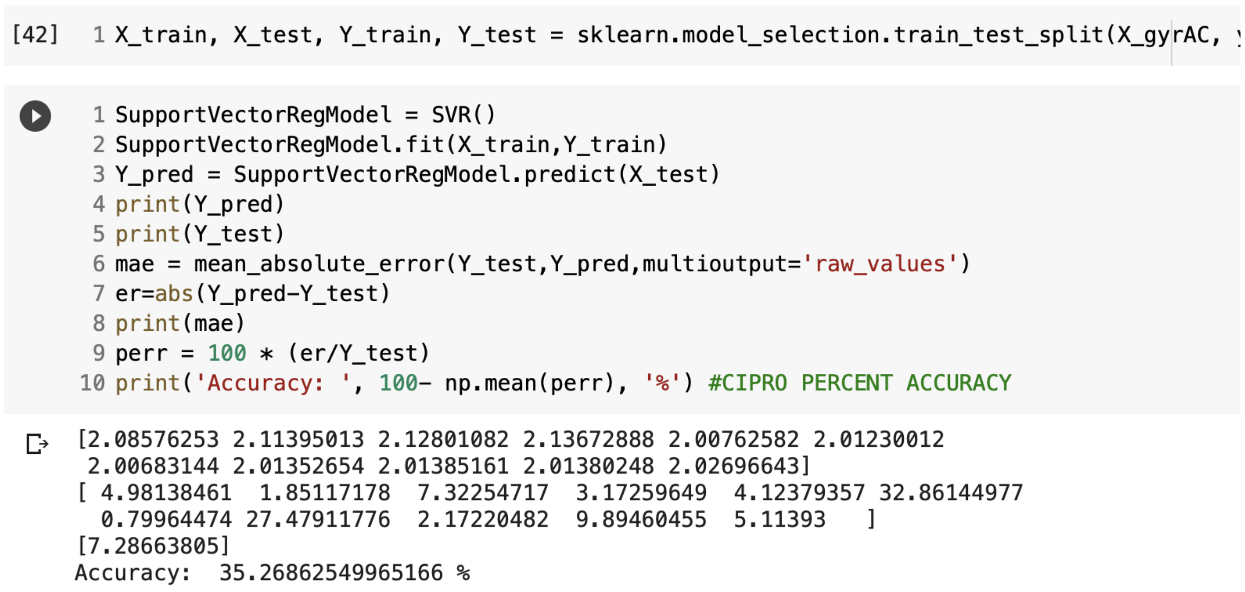


**Figure 1:** Comparing the Random Forest predictions (blue) and SVR predictions (red) to the actual MIC number (green).



```
[314]: print('Training Features Shape:', train_features.shape)
       print('Training Labels Shape:', train_labels.shape)
       print('Testing Features Shape:', test_features.shape)
       print('Testing Labels Shape:', test_labels.shape)

       Training Features Shape: (22, 28)
       Training Labels Shape: (22,)
       Testing Features Shape: (6, 28)
       Testing Labels Shape: (6,)

[315]: from sklearn.ensemble import RandomForestRegressor
       # Instantiate model with 1000 decision trees
       rf = RandomForestRegressor(n_estimators = 1000, random_state = 16)
       # Train the model on training data
       rf.fit(train_features, train_labels)

[315]: RandomForestRegressor(n_estimators=1000, random_state=16)

[318]: predictions = rf.predict(test_features)
       # Calculate the absolute errors
       errors = abs(predictions - test_labels)
       print(predictions)
       print(test_labels)
       # Print out the mean absolute error (mae)
       print('Mean Absolute Error:', round(np.mean(errors), 2))

       [2.87580718 4.25897317 5.8105842  3.70545577 1.1053376  3.26321117]
       [32.86144977  7.50993727 27.47911776  1.13333333  5.72817894  3.82608696]
       Mean Absolute Error: 10.44

[317]: # Calculate mean absolute percentage error (MAPE)
       mape = 100 * (errors / test_labels)
       # Calculate and display accuracy
       accuracy = 100 - np.mean(mape)
       print('Accuracy:', round(accuracy, 2), '%.')

       Accuracy: 10.71 %.
```

**Figure 2:** Random Forest modeling from Project 1. Using only the Percent Identity Matrix as feature evaluation.

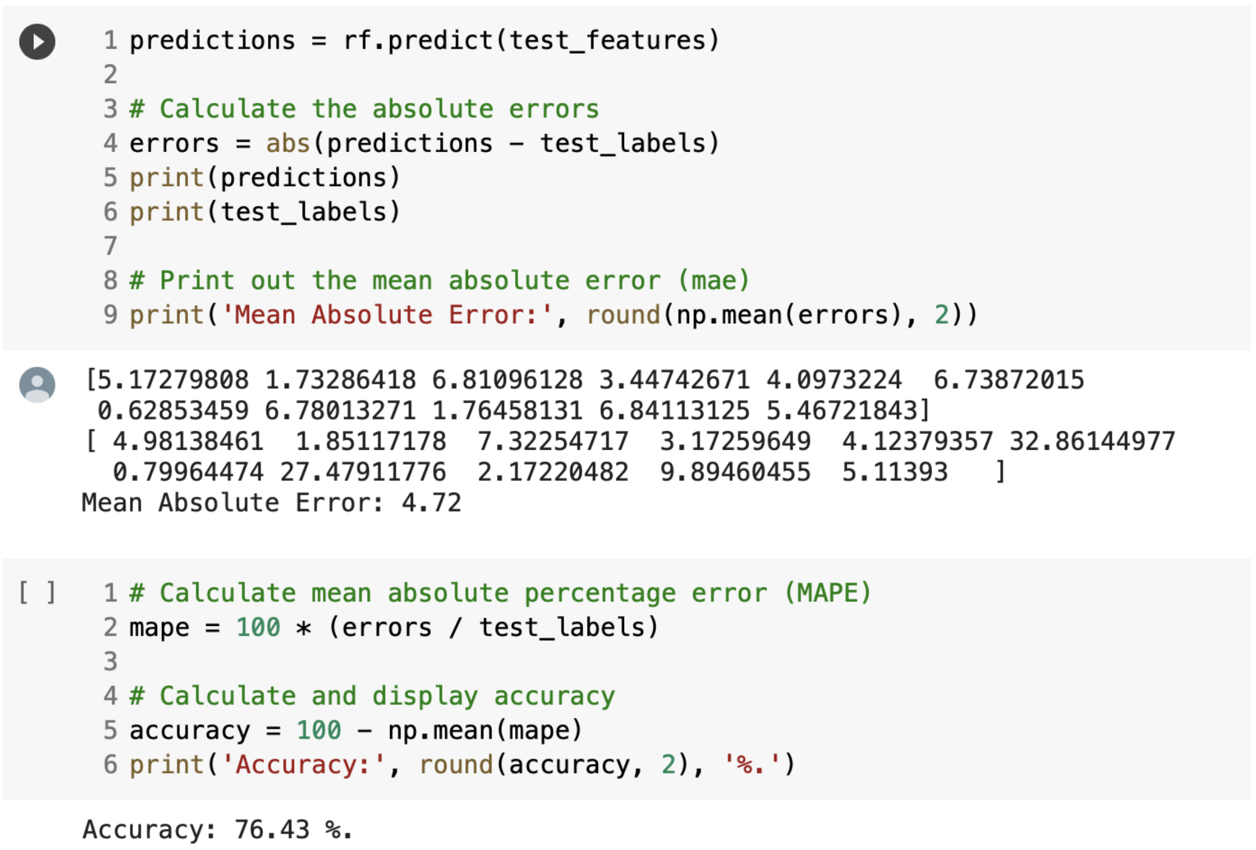


```
[42] X_train, X_test, Y_train, Y_test = sklearn.model_selection.train_test_split(X_gyrAC,
```

```
SupportVectorRegModel = SVR()
SupportVectorRegModel.fit(X_train,Y_train)
Y_pred = SupportVectorRegModel.predict(X_test)
print(Y_pred)
print(Y_test)
mae = mean_absolute_error(Y_test,Y_pred,multioutput='raw_values')
er=abs(Y_pred-Y_test)
print(mae)
perr = 100 * (er/Y_test)
print('Accuracy: ', 100- np.mean(perr), '%') #CIPRO PERCENT ACCURACY
```

```
[2.08576253 2.11395013 2.12801082 2.13672888 2.00762582 2.01230012
 2.00683144 2.01352654 2.01385161 2.01380248 2.02696643]
[ 4.98138461  1.85117178  7.32254717  3.17259649  4.12379357 32.86144977
  0.79964474 27.47911776  2.17220482  9.89460455  5.11393   ]
[7.28663805]
Accuracy:  35.26862549965166 %
```

**Figure 3:** Support Vector Regression model using k-mer econding of sequences.



```
predictions = rf.predict(test_features)

# Calculate the absolute errors
errors = abs(predictions - test_labels)
print(predictions)
print(test_labels)

# Print out the mean absolute error (mae)
print('Mean Absolute Error:', round(np.mean(errors), 2))
```

```
[5.17279808 1.73286418 6.81096128 3.44742671 4.0973224  6.73872015
 0.62853459 6.78013271 1.76458131 6.84113125 5.46721843]
[ 4.98138461  1.85117178  7.32254717  3.17259649  4.12379357 32.86144977
  0.79964474 27.47911776  2.17220482  9.89460455  5.11393   ]
Mean Absolute Error: 4.72
```

```
[ ] # Calculate mean absolute percentage error (MAPE)
mape = 100 * (errors / test_labels)

# Calculate and display accuracy
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')
```

```
Accuracy: 76.43 %.
```

**Figure 4:** Random Forest Regression modeling using k-mer encoding of sequences.

## References

1. Card R, Zhang J, Das P, Cook C, Woodford N, Anjum MF. 2013. Evaluation of an Expanded Microarray for Detecting Antibiotic Resistance Genes in a Broad Range of Gram-Negative Bacterial Pathogens. Antimicrob Agents Chemother. 57(1):458–465. doi:10.1128/AAC.01223-12.
2. Davies J, Davies D. 2010. Origins and Evolution of Antibiotic Resistance. MMBR. 74(3):417–433. doi:10.1128/MMBR.00016-10.
3. Dost B. 2010. Optimization algorithms for biological data. La Jolla: University of California, San Diego. [accessed 2021 Apr 15]. http://escholarship.org/uc/item/0s47h26s.
4. Drlica K, Zhao X. 1997. DNA gyrase, topoisomerase IV, and the 4-quinolones. Microbiol Mol Biol Rev. 61(3):377–392.
5. Festa P. 2007. On some optimization problems in molecular biology. Mathematical Biosciences. 207(2):219–234. doi:10.1016/j.mbs.2006.11.012.
6. Munita JM, Arias CA. 2016. Mechanisms of Antibiotic Resistance. Microbiology Spectrum. 4(2). doi:10.1128/microbiolspec.VMBF-0016-2015. [accessed 2021 Apr 16]. http://www.asmscience.org/content/journal/microbiolspec/10.1128/microbiolspec.VMBF-0016-2015.
7. Mustafa GR, Li C, Zhao S, Jin L, He X, Shabbir MZ, He Y, Li T, Deng W, Xu L, et al. 2021. Metagenomic analysis revealed a wide distribution of antibiotic resistance genes and biosynthesis of antibiotics in the gut of giant pandas. BMC Microbiol. 21(1):15. doi:10.1186/s12866-020-02078-x.
8. O'Shea K, Nash R. 2015 Dec 2. An Introduction to Convolutional Neural Networks. arXiv:151108458 [cs]. [accessed 2021 Apr 16]. http://arxiv.org/abs/1511.08458.
9. Park Y, Kellis M. 2015. Deep learning for regulatory genomics. Nat Biotechnol. 33(8):825–826. doi:10.1038/nbt.3313.
10. Patel S. 2014. Role of Proteomics in Biomarker Discovery. In: Advances in Protein Chemistry and Structural Biology. Vol. 94. Elsevier. p. 39–75. [accessed 2021 Apr 15]. https://linkinghub.elsevier.com/retrieve/pii/B9780128001684000032.
11. Reali F, Priami C, Marchetti L. 2017. Optimization Algorithms for Computational Systems Biology. Front Appl Math Stat. 3. doi:10.3389/fams.2017.00006. [accessed 2021 Apr 15]. http://journal.frontiersin.org/article/10.3389/fams.2017.00006/full.
12. Volz C, Ramoni J, Beisken S, Galata V, Keller A, Plum A, Posch AE, Müller R. 2019. Clinical Resistome Screening of 1,110 Escherichia coli Isolates Efficiently Recovers Diagnostically Relevant Antibiotic Resistance Biomarkers and Potential Novel Resistance Mechanisms. Front Microbiol. 10:1671. doi:10.3389/fmicb.2019.01671.

13. Wang Y, Zhang X-S, Chen L. 2010. Optimization meets systems biology. BMC Syst Biol. 4 Suppl 2:S1. doi:10.1186/1752-0509-4-S2-S1.
14. Whetton AD, Preston GW, Abubeker S, Geifman N. 2020. Proteomics and Informatics for Understanding Phases and Identifying Biomarkers in COVID-19 Disease. J Proteome Res. 19(11):4219–4232. doi:10.1021/acs.jproteome.0c00326.
15. Zankari E, Hasman H, Cosentino S, Vestergaard M, Rasmussen S, Lund O, Aarestrup FM, Larsen MV. 2012. Identification of acquired antimicrobial resistance genes. Journal of Antimicrobial Chemotherapy. 67(11):2640–2644. doi:10.1093/jac/dks261.
16. Zhang N, Juneau P, Huang R, He Z, Sun B, Zhou J, Liang Y. 2021. Coexistence between antibiotic resistance genes and metal resistance genes in manure-fertilized soils. Geoderma. 382:114760. doi:10.1016/j.geoderma.2020.114760.
17. Zhang X-X, Zhang T, Fang HHP. 2009. Antibiotic resistance genes in water environment. Appl Microbiol Biotechnol. 82(3):397–414. doi:10.1007/s00253-008-1829-z.
18. Zhang Y, Shen G, Hu S, He Y, Li P, Zhang B. 2021. Deciphering of antibiotic resistance genes (ARGs) and potential abiotic indicators for the emergence of ARGs in an interconnected lake-river-reservoir system. Journal of Hazardous Materials. 410:124552. doi:10.1016/j.jhazmat.2020.124552.