

6-15-2017

Machine Learning Offers Predictive Insight into the Silver Nanomaterial Protein Corona

Matthew Findlay

Follow this and additional works at: https://scholarcommons.scu.edu/bioe_senior



Part of the [Biomedical Engineering and Bioengineering Commons](#)

Recommended Citation

Findlay, Matthew, "Machine Learning Offers Predictive Insight into the Silver Nanomaterial Protein Corona" (2017). *Bioengineering Senior Theses*. 63.

https://scholarcommons.scu.edu/bioe_senior/63

This Thesis is brought to you for free and open access by the Engineering Senior Theses at Scholar Commons. It has been accepted for inclusion in Bioengineering Senior Theses by an authorized administrator of Scholar Commons. For more information, please contact rschroggin@scu.edu.

SANTA CLARA UNIVERSITY

Department of Bioengineering

I HEREBY RECOMMEND THAT THE THESIS PREPARED
UNDER MY SUPERVISION BY

Matthew Findlay, Daniel Freitas

ENTITLED

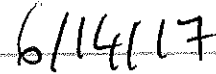
Machine Learning offers Predictive Insight into the Silver
Nanomaterial Protein Corona

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

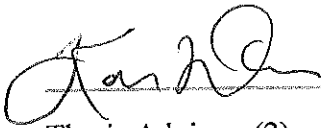
**BACHELOR OF SCIENCE
IN
BIOENGINEERING**



Thesis Advisor (1)



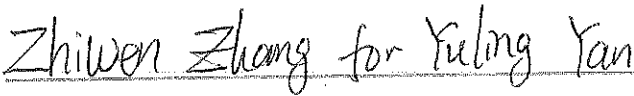
date



Thesis Advisor (2)



date



Department Chair



date

Machine Learning Offers Predictive Insight into the Silver Nanomaterial Protein Corona

by
Matthew Findlay, Daniel Freitas

Senior Design Project Report

Submitted to
the Department of Bioengineering
of
SANTA CLARA UNIVERSITY

in Partial Fulfillment of the Requirements
for the degree of
Bachelor of Science in BioEngineering

Santa Clara, California

2017

Machine Learning offers Predictive Insight into the Silver Nanomaterial Protein Corona

Matthew Findlay, Daniel Freitas

School of Engineering, Bioengineering

I) Abstract: The use of engineered nanomaterials (ENMs) in consumer and commercial products is increasing rapidly. The small size and high surface reactivity of ENMs gives them a range of attractive properties, and allows them to be incorporated into various materials. These properties make ENMs very appealing to modern industry, but also make ENMs toxic, causing serious health and environmental concerns. This toxicity is largely driven by the formation of a protein corona on the surface of ENMs. This protein corona is caused by proteins encountered in biological systems that bind to the surface of (ENMs). Despite the importance of the protein corona, little research has been done to control protein corona formation and model the biological conditions that contribute to protein-ENM binding. We present a quantitative characterization of proteins found bound on the surface of silver ENMs. A matrix of protein-ENM reactions were evaluated, including varied ENM sizes and surface coatings, as well as solution conditions (e.g. salt concentrations). Machine learning (random forest) classification was applied to this protein-ENM data matrix to evaluate the competing roles of the biophysical properties of proteins, ENM properties, and solution conditions in mediating formation of the ENM protein corona. The resulting model offers an accurate prediction of protein enrichment on ENMs with a receiver operating characteristic-score accuracy of 0.83. The effects of each variable in the formation of the ENM protein corona is calculated to provide recommendations for mechanistic models based upon protein quantification. Our model offers the framework to engineer ENMs to minimize the binding of toxic proteins, or maximize the binding of non-toxic proteins on the surface of ENMs.

Keywords: Engineered Nanomaterial, Protein Corona, Toxicity, Machine Learning, Mechanistic Model, LC-MS/MS Proteomics

II) Acknowledgements: We would like to dedicate our results to Dr. Wheeler of the Department of Chemistry and Biochemistry. We could not have succeeded without her biochemical expertise, constant support, and lab space. We would like to thank Dr. Mobed-Miremadi from the SCU Department of Bioengineering for her statistical expertise and unwavering support. We would like to thank Rich Eigenheer from UC-Davis, and Stanford's Mass Spectrometry Center for helping us develop our proteomics experiments. We would like to thank the National Institutes of Health, and the Santa Clara University Miller Center for their funding and support.



**National Institutes
of Health**

Grant: R15ES025929



Miller Center
for Social Entrepreneurship

Table of Contents

	page
Abstract.....	i
Acknowledgments.....	ii
1.0) Introduction.....	1
1.1) Background/Significance	1
1.2) Review of Literature.....	2
1.3) Statement of Project Goals and Objectives.....	3
1.4) Human and Environmental Health Concerns.....	3
1.5) Return on Investment	4
2.0) Methods.....	6
2.1) Yeast Protein Isolation.....	6
2.2) Protein-Particle Reaction.....	6
2.3) Mass Spectrometry.....	7
2.4) Database Development.....	7
2.5) Machine Learning Approach.....	9
2.6) Dimensionality Reduction.....	11
2.7) Model Validation.....	11
2.8) Comparison to other Algorithms	12
2.9) Assessing Feature Importance with Random Forests.....	12
3.0) Results and Discussion.....	13
3.1) Database.....	13
3.2) Dimensionality Reduction.....	16
3.3) Model Validation.....	16
3.4) Comparison with other Models.....	18
3.5) Quantification of Feature Importance.....	19

4.0) Conclusion.....	21
5.0) Bibliography.....	22

List of Figures

	<u>Page</u>
Figure 1: Total Global Nanotechnology Funding over time.....	5
Figure 2: Nanomaterial Product Lifecycle.....	5
Figure 3: The Experimental Database.....	8
Figure 4: Combining multiple decision trees to reduce overfitting.....	10
Figure 5: The Machine Learning Approach.....	11
Figure 6: Histogram of Protein Enrichment.....	14
Figure 7: RFECV Results.....	15
Figure 8: Confusion Matrix.....	18
Figure 9: Receiver Operating Characteristic Curve.....	19
Figure 10: Feature Weights.....	21
Figure 11: Clustered PI and Protein Weight.....	21

List of Tables

Table 1: Domain of Physicochemical features	16
Table 2: Validation Metrics.....	18
Table 3: Comparison with other Algorithms.....	22

List of Equations

Equation 1: Enrichment Factor.....	8
Equation 2: Gaussian Kernel.....	13
Equation 3: Logit Function.....	13

Appendix A

Source Code	29
--------------------------	----

Appendix B

Undergraduate Funding Request Letter	40
---	----

1.0) Introduction

1.1) Background/Significance

Engineered nanomaterials (ENMs) are a growing constituent of consumer and commercial products (figure 1¹) from transparent sunscreens and textiles to household and industrial cleaning supplies. Silver nanoparticles (AgNPs) in particular offer bacterial and UV resistance, semi-conductance and can be used as a deodorant. The small size and high surface reactivity of ENMs that elicit these attractive properties for use in modern industry can also render ENMs toxic and thereby pose serious health concerns. When nanoparticles come into contact with a biological system, proteins interact with and bind to the particle surface, forming what has come to be known as the protein corona². These biochemical surface changes, dominated by protein adsorption, play a large role in impacting ENM toxicity³, biomagnification⁴ and ecological fate⁵, ultimately altering the particles biological ‘identity’, and changing its expected physiological response^{6,7,8}. Due to these findings, and findings alike, our project aims to reduce ENM toxicity by creating a statistical framework for ENM-Protein interactions, with the hope of informing predictive models that can deliver accurate and vital information about ENM-protein reactivity and surface adsorption to industry partners involved in manufacturing and implementing ENM technologies. This statistical framework will offer a crucial step forward in solving design concerns related to the use release of nanomaterials into our global ecosystem.

As the use of ENMs by modern industry continues to increase, toxic ENM waste is released into the environment. This ENM waste spreads throughout our global ecosystem into waterways, airways, food supplies, residential areas, and eventually into humans (figure 2). Once exposed to the environment, ENM waste compounds, their relative toxicity increasing over time. The toxic effects of ENM pollution motivates our group to create a statistical framework that will aid in the

design of ENMs to reduce toxicity towards not only humans but the many layers of our global environment. Nanotechnology is a young field that is growing rapidly, and has the potential to be transformative in many areas of science and engineering, already leading to vast advancements in medicine^{9,16}, the food industry¹⁷, textiles, cosmetics and sports. Due to the powerful implications of nanotechnology, it is vital that steps are taken to ensure that ENMs are produced ethically to reduce human and environmental health concerns. While we understand that large scale environmental disasters occur on a daily basis, we argue that these large scale disasters distract from the tangible risks of small scale pollutants such as ENMs. In an age where technological advancements often take priority over human and environmental health, it is crucial that someone steps in to offer a statistical framework that will reduce the ethical concerns surrounding the use and release of ENMs. Although it is growing quickly, since nanotechnology is still a young field, we have a small window of opportunity to push the growth of the industry in an ethical direction before the field starts to grow at an uncontrollable rate.

1.2) Review of Literature

When studying the toxic effects of ENMs, researchers have failed to consider relevant biological conditions⁹. In other words, researchers have tested ENMs without consideration of how ENM chemistry changes within physiological systems. An example of this change is protein adsorption, which leads to the formation of a “corona” on ENMs, leaving the ENM surface with little resemblance to the original material^{6,7,8}, in essence changing its biological identity. The protein corona permanently alters ENM reactivity and by extension, also ENM toxicity^{2,10,11,12}. Although there is extensive evidence of nanomaterial toxicity^{13,14} and biomagnification^{4,5} in biological environments, little quantitative work has been done to explain the role proteins can play in altering nanomaterial fate. Our statistical framework is the first step towards a quantitative approach to a problem that has typically been dealt with through qualitative data evaluation. To offer a quantitative analysis, we employ machine learning to offer a robust predictive model and evaluate the importance of multiple variables. Machine learning has been

used to model nanomaterial datasets in the past¹⁵. However, we feel our analyses is unique because these models have failed to consider the effects of the protein corona.

1.3) Statement of Project Goals and Objectives

To address this issue and produce robust results, Python 2.0 and the scikit-learn package have been chosen to employ Random Forest Classification to generate our machine learning model¹⁸. Random forest classification was chosen as the predictive algorithm due to its relative insensitivity to outliers and noise, and ability to internally produce a list of feature importance. The power of a predictive model is limited to the quality of the dataset used to generate the model, therefore we employ a validated LC-MS/MS procedure previously developed in the Wheeler lab¹⁹ to acquire accurate and complete proteomics data. Our proteomics data quantifies the enrichment of proteins in unbound fractions as well as those bound tightly on the surface of nanomaterials. This enrichment data, coupled with protein characteristics taken from online databases and the ENM surface properties and solution conditions from our experimental procedure, gives our model the necessary data to produce a robust statistical framework that can offer strong predictive power into future nanomaterial interactions and offer insight into which nanomaterial characteristics can be altered to control protein adsorption. We are hoping that the same model we are developing for yeast protein can be applied also to ENM interactions with human protein populations and by extension other species as well. Hence the research objectives of our project are twofold: (i). To rank the statistical significance of factors relevant to protein-enrichment and (ii). To construct a classifier model based on the Random Forest algorithm to predict the Protein Corona formation under equilibrium binding conditions.

1.4) Human and Environmental Health Implications

As the use of ENMs by modern industry increases, toxic ENM waste is released into the environment. This ENM waste spreads throughout our global ecosystem into waterways, airways, food supplies, residential areas, and eventually humans (Figure 2). Once exposed to the global ecosystem, ENM waste compounds, increasing in toxicity over time. The toxic effects of

ENM pollution motivates our group to aid in the design of ENMs to reduce toxicity towards not only humans but the global ecosystem. Nanotechnology is a young field that is growing rapidly, and has the potential to completely transform science and engineering. Due to the powerful implications of nanotechnology, it is vital that steps are taken to ensure that ENMs are produced ethically to reduce human and environmental health concerns. In an age where technological advancements often take priority over human and environmental health, it is crucial that someone steps in to offer a statistical framework that will reduce the ethical concerns surrounding the use and release of ENMs. Although it is growing quickly, nanotechnology is still a young field. We still have a small window of opportunity to push the growth of Nanotechnology in an ethical direction before the field starts to grow at an uncontrollable rate.

More specifically, the fate and transport of engineered nanomaterials (ENMs) in the biota is mediated by proteins that coat ENMs in a protein corona (PC). An array of in depth experimental studies have provided characterization of the ENM PC for various organisms and conditions, establishing the importance of PCs; yet, in each new system, a costly PC characterization must be performed. The random forest classification approach developed here-in models PC populations for an array of ENM properties and reaction conditions, while providing insight into feature importance to define which aspects of protein, ENM, and solvent chemistry are most important to defining the PC population. The model can be widely applied and the approach represents the first step toward a predictive model for ENM PC populations.

1.5) Return on Investment

Investing in our project offers more than just a clean environment and safe world for your children. Each year, the National Institute of Health spends 14.5 Billion Dollars on animal testing²⁰. Our project offers an alternative method to animal testing by modelling toxicity pathways. As computational power increases, more toxicity tests will move to computational modelling as it is cheaper, offers a deeper insight into the mechanisms of toxicity, and does not harm animals. As the field of toxicity testing shifts to computational means, so will the billions of dollars associated thereof. We can use our software design to model other toxicity

mechanisms, and have the potential to be one of the first companies to take advantage of this paradigm shift. Although our project is deeply rooted in scientific inquiry, it will still produce tangible products. Once we finish modelling the protein corona that forms in biological systems, we aim to engineer a protein corona that can be pre-applied to ENMs before they are incorporated into nano-enabled products (NEPs). The pre-application of a protein corona can inhibit undesired protein binding to ENMs when they enter biological environments, giving manufactures full control over the fate of ENMs. This protein corona can be patented and sold to Nanotechnology manufacturers. If our engineered protein corona is truly effective and minimizes ENM toxicity, we predict that it will be required by the EPA for all NEPs. As of right now, there have been no attempts to engineer a protein corona, and there are no products that can control ENM fate. Our project has no competition and the potential to create a massive return on investment.

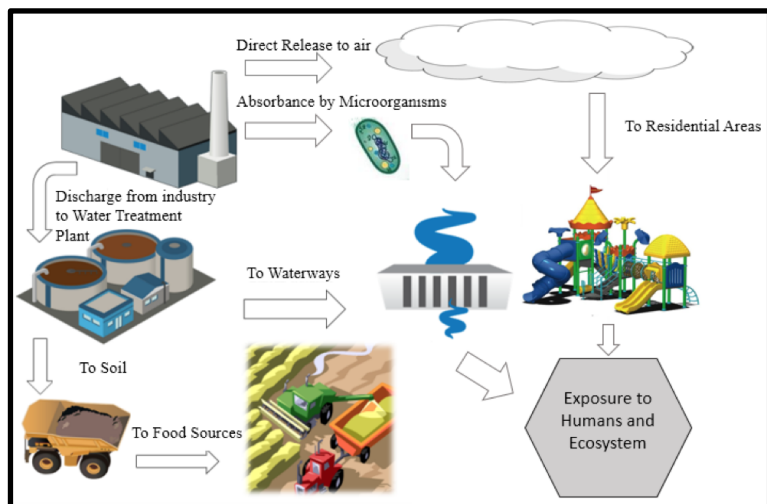


Figure 2: A schematic of how ENMs are spread from industry into the ecosystem, impacting human and environmental health. (Modified from citation 21)

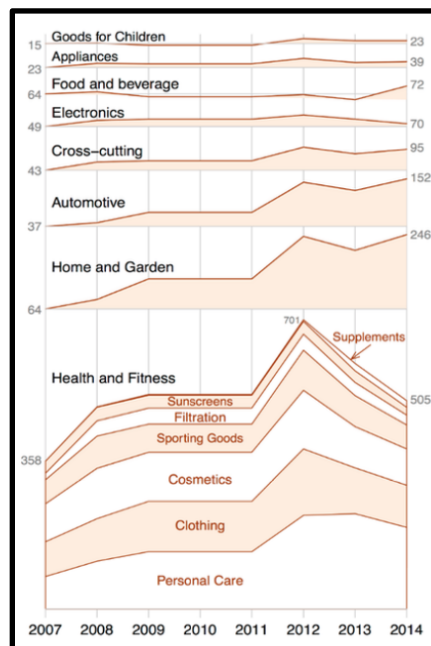


Figure 1. ENM use over time.

2.0) Methods

2.1) Yeast Protein Isolation

2L of BY4 yeast was cultured at 30C up to an optical density of 1 at $\lambda = 595$ nm, and was then harvested by centrifugation (Alegra X-22R) at 2300 x g for 30 minutes at a time. Cell pellets were washed with ice cold water and resuspended in 50 mM Ammonium Bicarbonate (AmBic, Sigma), pH 7.4. The resuspended cells were added dropwise to liquid nitrogen and mechanically lysed with mortar and pestle. As soon as the cells had been lysed, 50 mM AmBic containing 15 uL of protease inhibitors (Halt Protease Inhibitor Cocktail, Sigma Aldrich) was added to prevent destruction of solubilized proteins. Cell debris was removed by centrifugation (15 min, 3900 x g) and the remaining supernatant (containing our yeast protein population) was moved to a clean falcon tube. Protein extract was dialyzed, 3 mL at a time, in 1L of 50 mM AmBic, pH 7.4, with 3500 MWCO dialyzers and constant stirring. Protein concentration was determined with a BCA protein assay kit (Thermo Scientific), and SDS page was performed to confirm that our sample contained protein and not peptides.

2.2) Protein-Particle Reaction

We allowed 0.25 mg mL⁻¹ of silver nanoparticles (AgNPs, Nanocomposix) to react with 0.21 mg mL⁻¹ of solubilized yeast protein overnight at 37C for each experiment in our matrix, in triplicate. AgNPs and the proteins strongly associated to their surface were removed from the unassociated proteins in solution by centrifugation (15000 rpm, 30 min). This step was repeated, washing with 50 mM Ambic between spins, to ensure that only the proteins most tightly bound to the particle surface remained, and that those enriched in solution were removed. The removed supernatant was placed in eppendorfs labeled 'unbound' and the particles spun from solution were suspended in 50 mM AmBic and labeled 'bound.' The unbound protein samples were concentrated in 3000 MWCO millipore concentrators by centrifugation at 14000 rpm for 10 min, a total of four times. At this point we have two sets of triplicate samples, one set for unbound proteins and one set for bound proteins.

To reduce disulfide bonds, 5 μ L of 200 mM TCEP (Sigma) was added to each sample, which were then vortexed, spun down and left to reduce at room temperature for 1h. Sulfhydryl alkylation was done by adding 4 μ L of 1 M iodoacetamide (Sigma) in 10 mM AmBic left at room temperature for 1h, before being neutralized by adding of 20 μ L of TCEP and incubated for an additional hour. Samples were incubated at 37° C overnight after addition of 2 μ L of mass-spec grade trypsin (promega), in order to digest the proteins in unbound solution and on the particle surface. After digestion, any peptide in both the unbound and bound triplicate sets were separated from any remaining AgNP debris by centrifugation (15000 rpm, 15 min). Samples were packaged and sent to Stanford University Mass Spectrometry Center on dry ice for LC-MS/MS proteomic analyses.

2.3) Mass Spectrometry

All LC-MS/MS analyses was performed at Stanford University's mass spectrometry center. Digested peptides were suspended in 0.1% trifluoroacetic acid and 2% acetonitrile by centrifugation (15000 rpm, 10 min) and sonication. Peptides were then analyzed by LC-MS/MS using a Thermo LTQ ion trap mass spectrometer with a nano-spray source after separation at 2 μ L min⁻¹ via 200 μ m x 150 mm C18 reversed phase column. The two buffer mobile phase system included 0.1% formic acid and acetonitrile, with a 2h long gradient. MS/MS spectra were acquired and an MS survey scan was obtained with a m/z range of 375-1400. As previously stated, three replicates were used for both the bound and unbound protein sample sets. Peptide identification by this method was accepted if there presence was determined with over 85% probability.

2.4) Database Development

Protein abundance and spectral counts were obtained from LC-MS/MS. To obtain protein enrichment, spectral counts were divided by protein length and then normalized in the database as seen in (eq 1). The ratio of bound NSAF values to unbound NSAF values were taken for each

protein-particle pair to obtain an enrichment factor. Enrichment factors greater than one were considered bound, and enrichment factors less than one were considered unbound. For each protein identified by MS proteomics, biophysical characteristics were obtained from Uniprot²², including molecular weight, pI, IP number, and amino acid sequence. Interpro numbers²³ were also included when available for a protein. ENM characteristics were assigned based upon experimental characterization. This includes ENM size rounded to 10 or 100 nm and zeta-potential assigned as a binomial (either negative or positive). Finally, solvent conditions were summarized as two categories. The first category was either 0, 0.8, or 3.0 mM NaCl, while the second category included either 0 or 0.1 mM cysteine. The database was organized as seen in **Figure 3**.

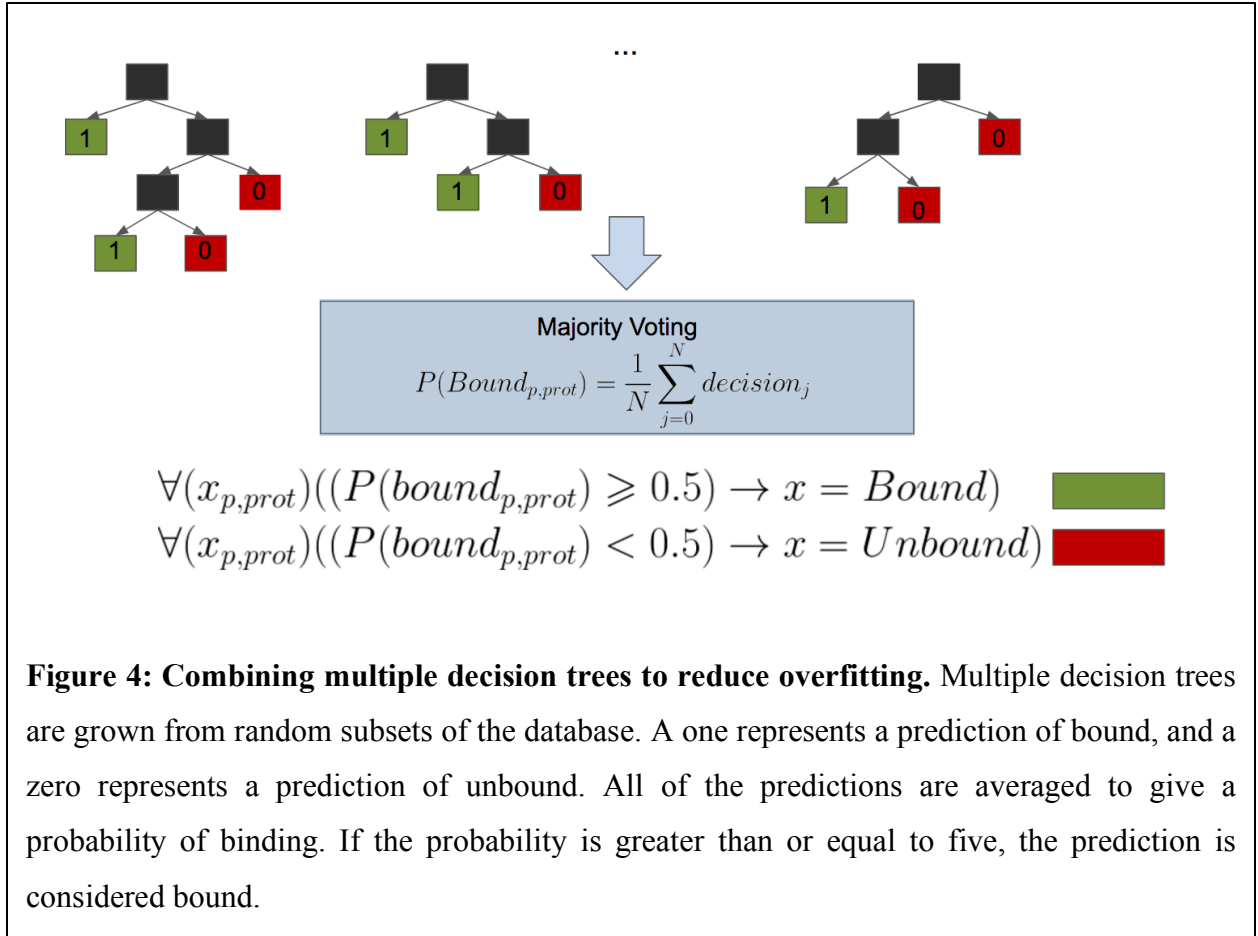
$$NSAF_N = \frac{S_N/L_N}{\sum_{i=1}^n s_i/L_i} \quad [1]$$

Observations	feature 1	feature 2	... feature m	Classification Target
Experiment 1	Value or Class	Value or Class	Value or Class	Bound/Unbound
Experiment 2				
.				
.				
.				
Experiment n				

Figure 3: Database structure.

2.5) Machine Learning Approach

Python was used to employ Random Forest Classification (RFC) with the scikit-learn package. RFC was chosen because it is a robust ensemble learning method that combines multiple decision trees to form a predictive model that is less susceptible to overfitting than traditional decision trees, as seen in **Figure 4**. Similar approaches have proven successful in analysis of other proteomics datasets²⁴ and other predictions of ENM fate²⁵. Each decision tree produces a predictive model by splitting data using simple decisional rules.²⁶



RFC then returns the majority vote produced by the group of predictive models. Our implementation of RFC can be summarized into five steps, summarized in **Figure 5**: (1.) Each protein-particle pair in the database was represented as a vector containing each feature as a normalized dimension. (2.) 90 % of the dataset was randomly partitioned from the database to

train the model, leaving 10% of the data to test the model. (3.) 10,000 random bootstrap samples of size $\log(n)$ were drawn from the testing partition and a decision tree was grown from each sample, (4.) The predictions produced by each tree were aggregated and used to classify proteins as PC or non-PC based on the majority vote between the trees. (5.) This process was repeated 50 times, each time with a newly selected random database partition to avoid bias. Majority voting between trees reduces the risk of overfitting as the decision trees containing outliers and noise will be outnumbered by the rest of the decision trees during the voting process.

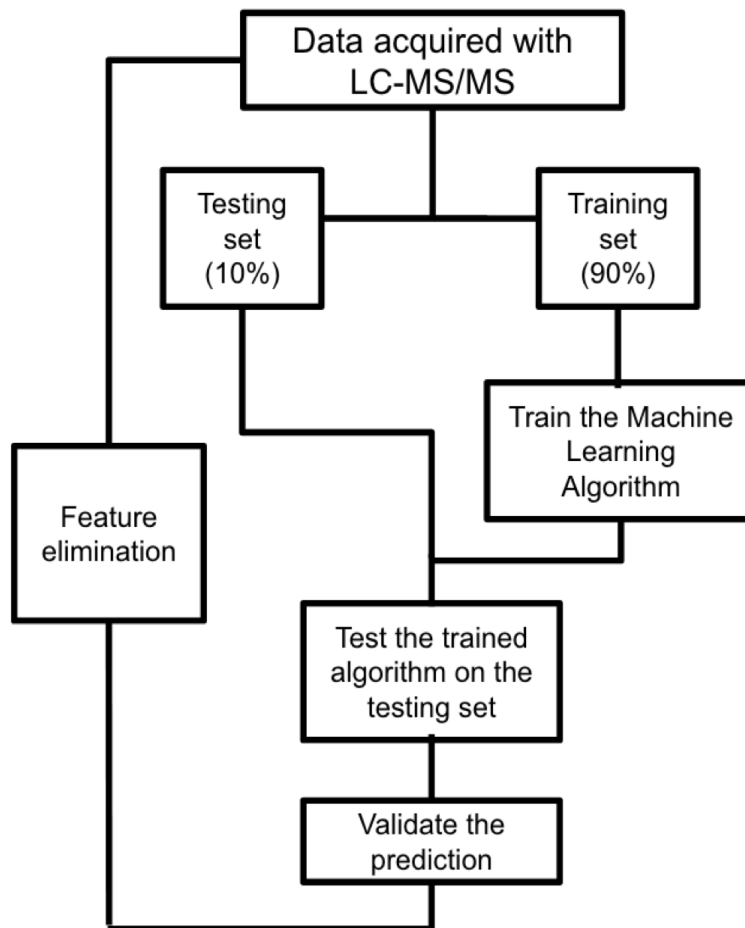


Figure 5: Graphical depiction of the data partitioning scheme and model structure.

2.6) Dimensionality Reduction

To remove noise from the dataset Recursive Feature Elimination and Cross-Validation was employed (RFECV) using the criteria of mean decrease impurity. The mean decrease impurity is defined as the weighted probability of reaching a node in a decision tree averaged over all of the 10,000 trees.

RFECV is an adopted dimensionality reduction algorithm based on the idea of recursively constructing a model, choosing the least important variable (based on average decrease impurity), removing the variable, and reconstructing the model. At each iteration, 5-fold cross-validation is conducted to determine the predictive power of the model. The iteration with the best power contains the optimum number of features to train the model.

2.7) Model Validation

To give a clear and un-bias validation of our model, several validation metrics common in the fields of biostatistics and machine learning were employed. These metrics include precision, recall, F1-score, area under the receiver operating characteristic curve (AUROC),²⁷⁻³⁰ and accuracy. In a binary decision problem, a classifier labels data as either positive or negative. In this case, positive means that a protein will be part of the PC, and negative means the protein will be Non-PC. This gives our classifier four possible outcomes: (1.) A protein is properly classified as PC (True positive). (2.) A protein is improperly classified as PC (False Positive). (3.) A protein is properly classified as Non-PC (True Negative) (4.) A protein is improperly classified as Non-PC (False Negative). These four possible outcomes can be counted and summarized using our validation metrics. Recall is the number of true positives divided by the total PC-proteins in the dataset. Precision is the number of true positives divided by the sum of true positives and false positives produced by the model. The F1-Score is simply the harmonic mean of precision and recall. Accuracy is the number of true positives and true negatives divided by the total number of classifications made by the model. The ROC curve shows how the number of true positives varies with the number of false positives produced by the model at different

cutoffs. The AUROC is the area under the ROC curve, AUROC is typically reported as it gives a normalized score between 0 and 1 produced by the ROC curve.

2.8) Comparison to other Algorithms

Support Vector Machines (SVM)³¹ and Logistic Regression (LR)³² were employed along with the RFC algorithm on the dataset to produce a well-rounded understanding of the predictive power that could be generated from database. SVM and LR were chosen due to their extensive use in the fields of biostatistics and machine learning. SVM was employed for classification with a radial basis function kernel known as the Gaussian kernel [equation 2], and binary LR was fit with a logit model [equation 3]. Both models performed well on the dataset suggesting that future work may benefit from the use of several machine learning algorithms in ensemble fashion.

$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right) \quad \vdots \quad [2]$$

$\|x - x'\|^2$ *Squared Euclidean distance between training set vectors*
 $2\sigma^2$ *Spread parameter*

$$\log\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \beta_1 x_i \quad [3]$$

p_i *probability of binding based on NSAF cutoff*
 x_1 *binary feature variable*

2.9) Assessing feature importance with random forests

A measure of variable importance was calculated as the mean decrease impurity in the 10,000 implemented decision trees.³³

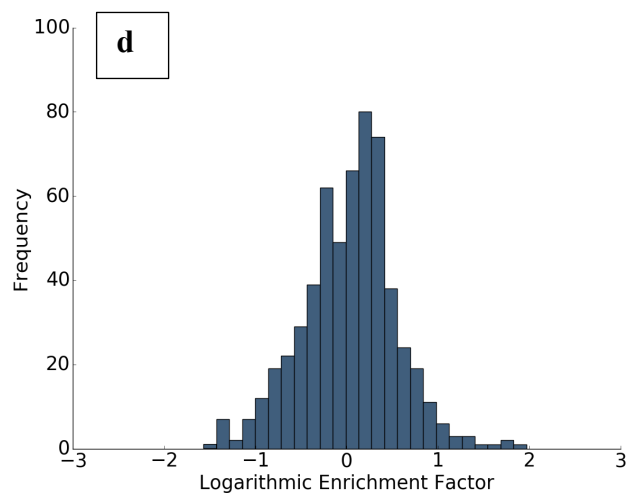
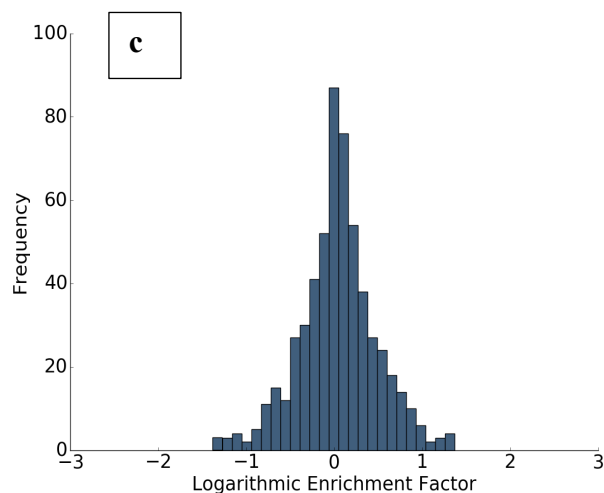
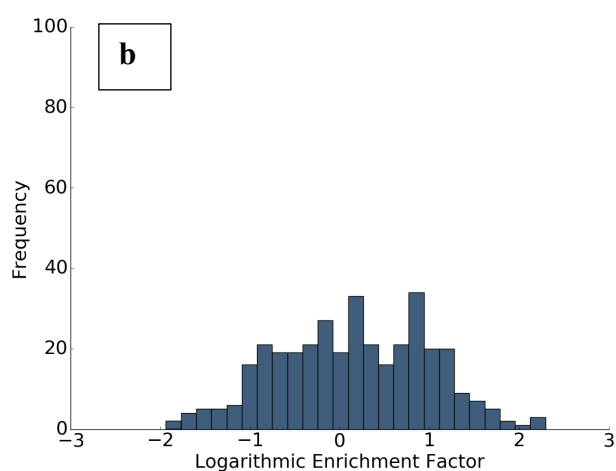
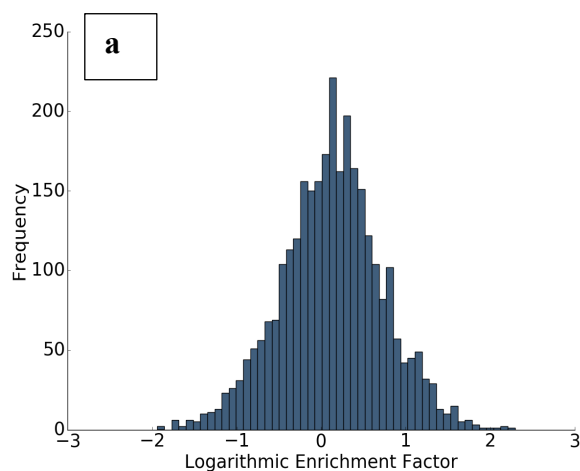
3.0) Results and Discussion

3.1) Database

A proteomics produced database of yeast protein enrichment on silver (Ag) ENMs was used for the machine learning model because of the ubiquity of yeast in the environment, widespread use of Ag ENMs in consumer products, and extensive set of proteins characterized in the database. The Ag ENM PC database includes 1414 yeast proteins characterized for enrichment on Ag ENMs as detailed in Eigneer *et al.* Protein enrichment was evaluated by the log of the ratio of protein abundance in solution and on an Ag ENM, resulting in enrichment factors that are positive for proteins enriched on Ag ENMs and negative for those enriched in solution. A total of 3012 protein enrichment values were recorded. Within the Ag ENM PC database used for this study, the majority of proteins show weak enrichment in solution or on ENMs and few are strongly enriched in either population. Logarithmic Enrichment factors across all the particles were plotted as a histogram (**Figure 6**). Across the entire dataset, the enrichment data forms a relatively Gaussian distribution. This is as expected, since the majority of proteins are not expected to have strong enrichment in PC or solution. Yet, there are many proteins still present in the tails of the distribution with strong enrichment in PC or solution. Importantly, when the histogram is plotted for each individual sample, trends are clearly visualized. For example, the histogram for positively and negatively charged Ag ENMs with no cysteine or NaCl are significantly different, where the positively charged Ag ENMs have a smoother distribution of enrichment than those with a negative surface coating. Yet when solvent conditions are changed, the distribution of the negatively charged ag ENMs become more similar to the positively charged Ag ENMs.

For each yeast protein evaluated for enrichment, nine physiochemical features/parameters were recorded, along with solvent features (2 levels) and Ag ENM (2 levels) characteristics. The experimental variables comprised in the nine training features are listed in **Table 2** with the corresponding range of each feature. Because of the large number of proteins examined (N=3013) it can be assumed that the logarithmic enrichment factors and other protein properties

are randomly-distributed across the experimental database. By comparison to protein features, ENM and solvent properties are underrepresented in the model training features. Variations in ENM and solvent. properties are more difficult to explore, because expansion of these variables in the experimental matrix requires a new protein-ENM reaction and proteomics runs.



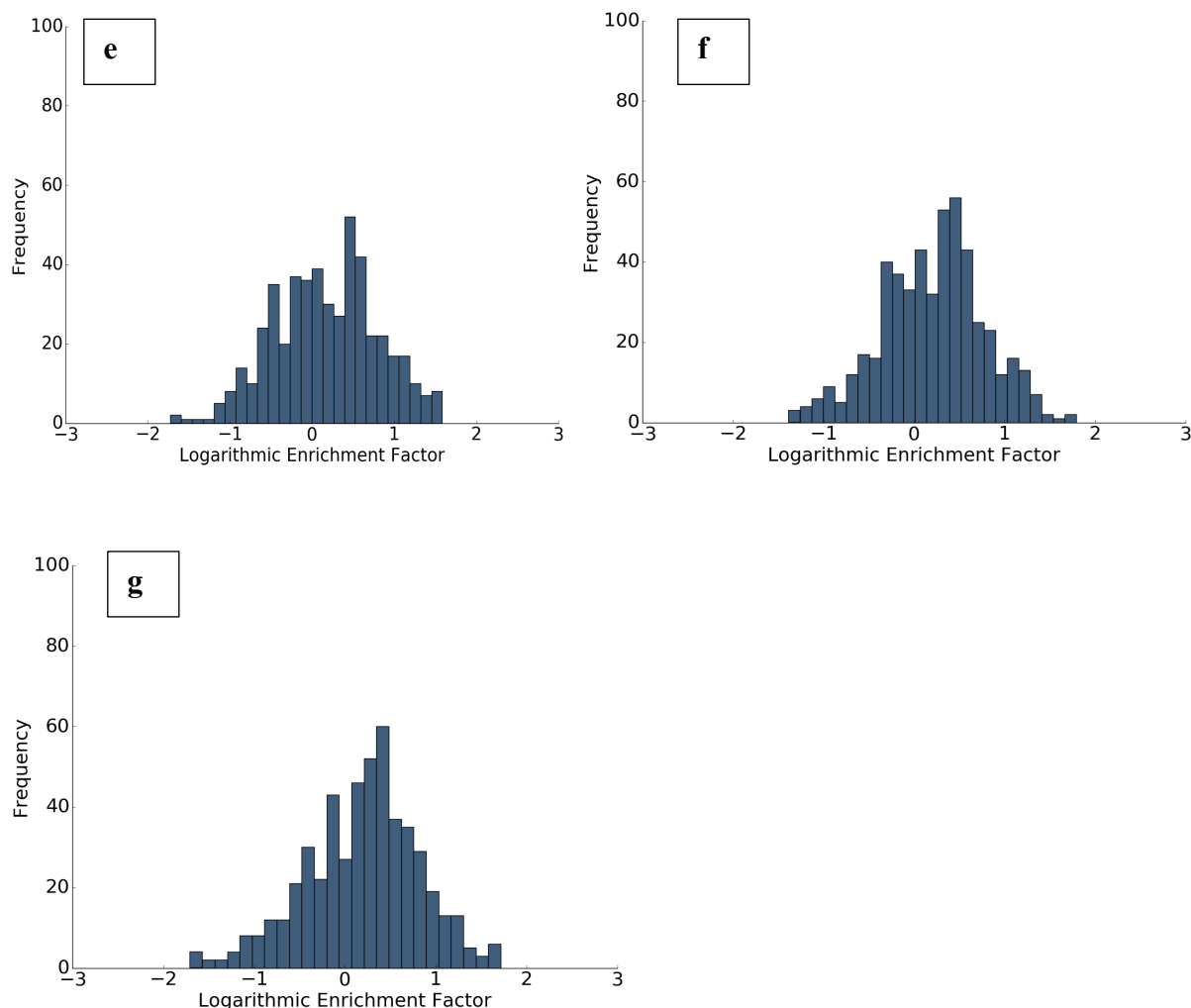


Figure 6: Histogram of logarithmic enrichment factors for all proteins across the range of ENMs and solution conditions within the database. The histogram is shown with 50 bins. The enrichment from all datasets is shown in (a). The enrichment distribution for each individual sample is shown as follows: (b) cationic branched polyethyleneimine coated 10 nm Ag ENMs in 10 mM sodium phosphate pH 7.4, (c) anionic citrate coated 10 nm Ag ENMs in 10 mM sodium phosphate pH 7.4, (d) anionic citrate coated 100 nm Ag ENMs in 10 mM sodium phosphate pH 7.4, (e) anionic citrate coated 10 nm Ag ENMs in 10 mM sodium phosphate pH 7.4 with 0.1 mM cysteine, (f) anionic citrate coated 10 nm Ag ENMs in 10 mM sodium phosphate pH 7.4 with 0.8 mM sodium chloride, and (g) anionic citrate coated 10 nm Ag ENMs in 10 mM sodium phosphate pH 7.4 with 3.0 mM sodium chloride.

Training Features	Range within dataset (method of determination)
<i>Protein characteristics</i>	
Isoelectric point	3.77 to 12.55
Protein weight	6 to 559 <u>kDa</u>
protein abundance	$10^{-7.40}$ - $10^{-4.17}$
% positive amino acids	4.72-39.00
% negative amino acids	0-33.33
% hydrophobic amino acids	13.80-60.66
% aromatic amino acids	0-11.86
% cysteine	0-7.14
<u>InterPro</u> numbers	range of 50
<i>ENM characteristics</i>	
ENM size	10 nm and 100 nm
ENM surface charge	Positive (+) and Negative (-)
<i>Solvent characteristics</i>	
Cysteine concentration	0, 0.1 <u>mM</u>
<u>NaCl</u> concentration	0, 0.8 <u>mM</u> and 3.0 <u>mM</u>
Target features	
Protein corona (PC) or not (non-PC)	PC or non-PC

Table 1 Domain of physicochemical features within the training and target dataset used for the machine learning effort.

3.2) Dimensionality Reduction

To remove features with no predictive value from the dataset, recursive feature elimination and cross-validation was employed (RFECV). Although originally included in the model in response to suggestions from Rihn and Joubert³⁴, protein InterPro numbers were eliminated from the model by the RFECV analysis **Figure 7**. With this elimination, data dimensionality was reduced from thirteen to twelve dimensions, including biophysicochemical features of the proteins, ENMs, and solvent (vide infra).

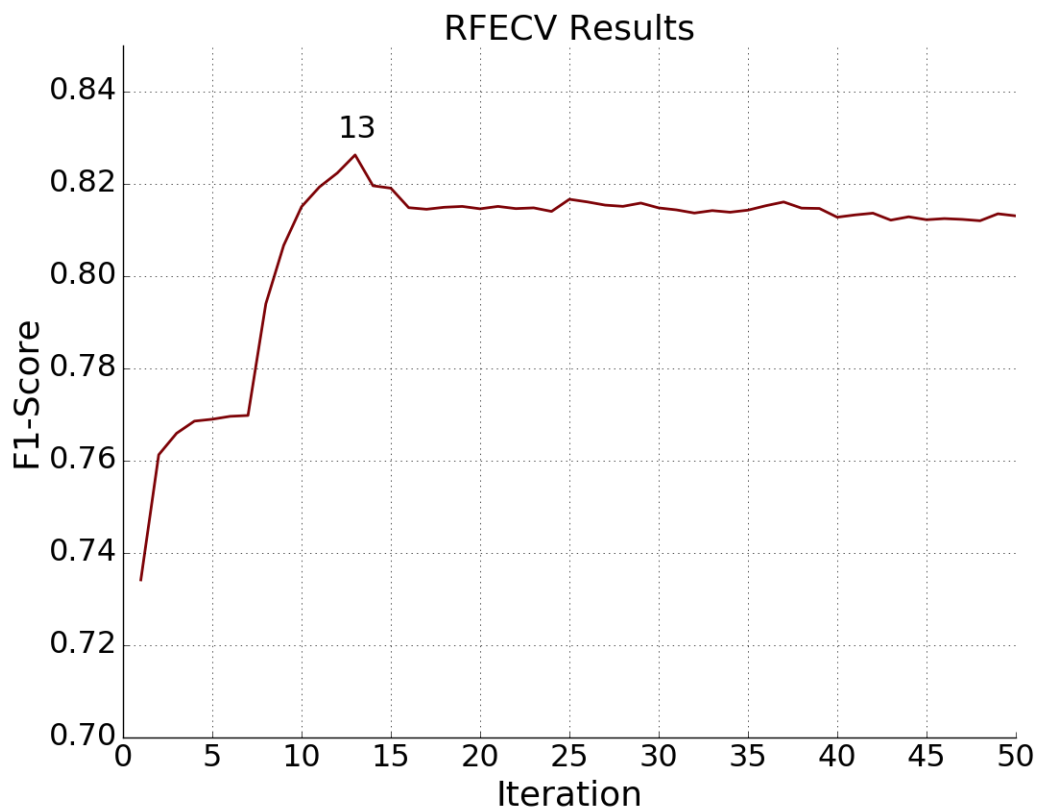


Figure 7: RFECV Results. Categorical variables were represented as dummy variables, increasing dimensionality to 50. Variables of least importance were deleted iteratively. The best performance was found with 13 dummy variables.

3.3) Model Validation

To validate the model, standard machine learning metrics were used, including precision, recall, accuracy, and the F1-score. Validation metrics are summarized in **Table 2** along with a raw confusion matrix **Figure 8**. Precision and recall are widely used performance metrics that offer a well-rounded evaluation of predictive performance. Model precision is 0.77 ± 0.02 , indicating that 77 % of the PC assignments made by the model were truly PC proteins. Recall is 0.85 ± 0.02 . In other words, 85% of the PC proteins in the dataset were predicted as PC. The F1-Score, the harmonic mean of precision and recall, was 0.81 ± 0.02 for this model. With an accuracy of 0.76 ± 0.02 , the model has a good predictive power for both PC proteins and non-PC proteins.

To further model validation, a receiver operating characteristic (ROC) curve was plotted with 302 decisional thresholds based on the models outputted probability of binding (**Figure 9a**). Generally, the convex shape of the ROC curve indicates a higher true positive rate at the expense of relatively lower false positive rate. In other words, the likelihood of correctly classifying a protein as PC is high, while incorrect classifications of PC are low. The area under the receiver operating curve (AUROC) for the resulting model is generally considered indicative of the predictive power of the model²⁷⁻²⁹ and can be interpreted as the model's ability to correctly classify proteins as PC or non-PC. With an AUROC of 0.83, the model performs significantly higher than the value of 0.5 for a random guess curve. More specifically, AUROC scores are evaluated relative to the complexity of the classification task. As the first to test this approach on ENM PC predictions, this work establishes a baseline of AUC performance for future predictive models. To provide a comparative metric for a problem of similar complexity, protein-protein binding predictions, Sain et al³⁵ report an AUCROC score of 0.7, which is typically considered strong for problems of this complexity. Related, the Youden index defines the threshold in the ROC curve that gives the best performance (**Figure 9b**). The optimum threshold was found to be 0.5, as seen by the maximum Youden index value. In other words, our ensemble method performs as expected, where the most proteins are properly assigned as PC when 50% or more decision trees assign the protein as PC.

Metric	Score	Equation
Recall	0.85 ± 0.02	$= \frac{TP}{(TP + FN)}$
Precision	0.77 ± 0.02	$= \frac{TP}{(TP + FP)}$
F1-Score	0.81 ± 0.02	$= \frac{2 * Recall * Precision}{(Recall + Precision)}$
Accuracy	0.76 ± 0.02	$= \frac{(TP + TN)}{(TP + TN + FP + FN)}$

Table 2: Validation metrics, scores, and accompanied equations.

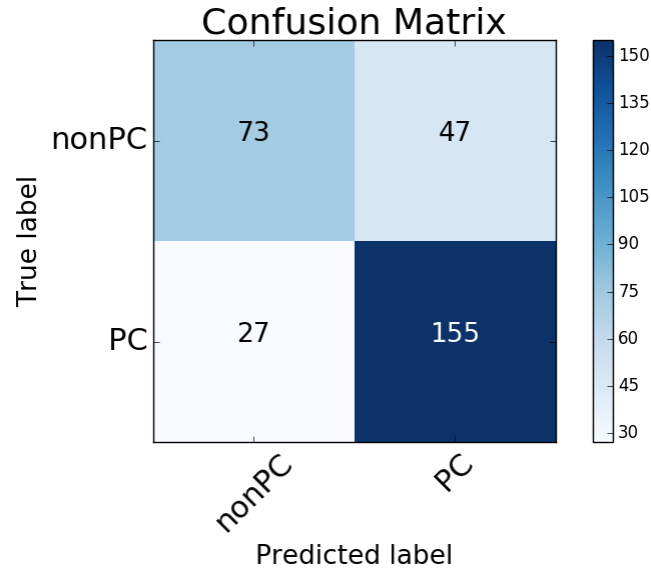


Figure 8: Raw confusion matrix with the average predicted result when the model runs 50 times.

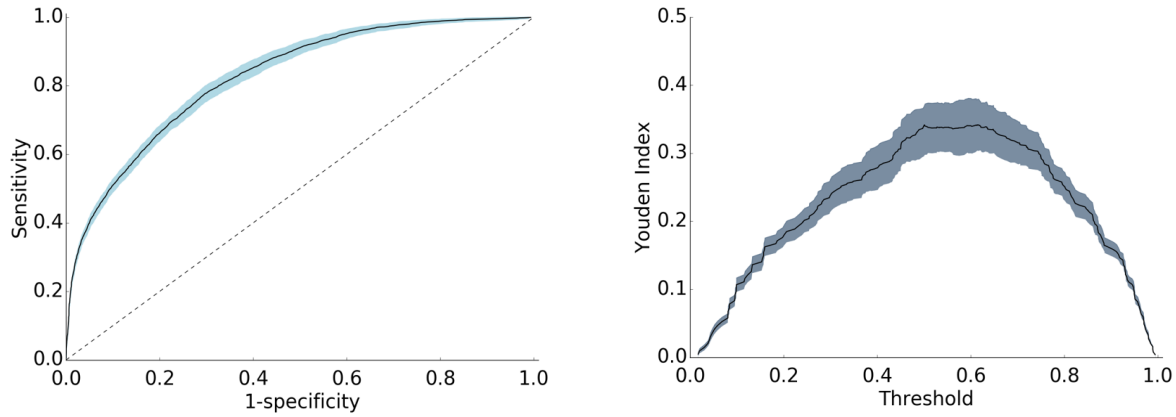


Figure 9: Receiver operating curve (a) and Youden index curve (b) for the final model. The receiver operating curve for the model (a) is shown with a solid line, error bars are in light blue, and the random guess curve is shown with a dashed line. The area under the curve for the receiver operating curve (AUROC) is 0.83. The Youden index curve (b) for the model is shown with a solid black line and error bars in grey-blue. The threshold for the maximum Youden index is 0.5.

3.4) Comparison with other models

Two other well-established algorithms were employed to provide a comparative for the RFC approach. Logistic regression (LR) and support vector machines (SVM) were chosen because of their extensive use in the fields of artificial intelligence and biostatistics.^{28,29,30} The SVM and LR algorithms were trained and tested with the same method as the random forests. The F1-score for these methods are 0.80 and 0.75, respectively, as summarized in Table 3. A lower F1-score for LR is not surprising, as LR fits to data points as if they are along a continuous function and therefore doesn't perform well when the feature space is too large and there is a large number of variables. Although SVM performed similarly to RFC, we still prioritize use of RFC in this context because it includes intuitive decision rules and is more readily accessible. It is possible, however, that future work with an expanded dataset, including significantly more features and observations, will perform better with the SVM algorithm.

Model	F1-Score
Random Forest Classification	0.81
Support Vector Machines	0.80
Logistic Regression	0.75

Table 3: The discriminatory capability of each model. Random Forest Classification and Support Vector Machines both have strong performance, Logistic Regression does not perform as well as the other classifiers.

3.5) Feature Importance Results

In addition to its ability to take into account variable interactions, RCF is useful because it can provide a measure of hierarchical variable importance. These feature weights give insight into the variables of importance in predicting and controlling ENM-protein interactions. Feature importance is shown in **Figure 10**. The pareto indicates that protein biophysical characteristics are more strongly weighted than solvent and ENM characteristics within the model and that protein characteristics dominate PC formation. The comparative sample for ENM and solvent characteristics is simply too small to derive conclusions across protein, ENM, and solvent features. Relative importance within each of these three feature sets, however, are useful to compare.

Among protein features, factors contributing to protein charge, including pI and percent of positively and negatively charged amino acids, together make-up nearly 50 % of feature importance. This reinforces earlier studies qualitatively reporting the importance of protein charge in PC formation^{16,19}: Long-range electrostatic interactions drive initial protein-ENM interactions playing a significant role in the stability of the hard corona. The slightly higher weight of salt concentration over cysteine within solvent features again points to the importance of electrostatics in PC formation. Our data also supports that proteins with higher pI values tend to bind to ENMs, as seen in **Figure 11**.

Across ENM features examined, ENM size and surface charge are weighed nearly evenly. This result is somewhat surprising, given the importance of electrostatics within the array of protein features; however, it indicates the importance of contributions from other features in model accuracy. Notably, within protein features, percentage of hydrophilic and aromatic amino acids, along with cysteine, also contribute significantly at nearly 25 %, indicating that hydrophobic interactions and perhaps metal-thiol bonds play a secondary role in the stability of the hard corona. As previously discussed, there is some selectivity for molecular weight within the PC^{4,6}. This data supports correlations between ENM and protein size and contributes to the hypothesis that decreased curvature of large ENMs may more easily support larger proteins.

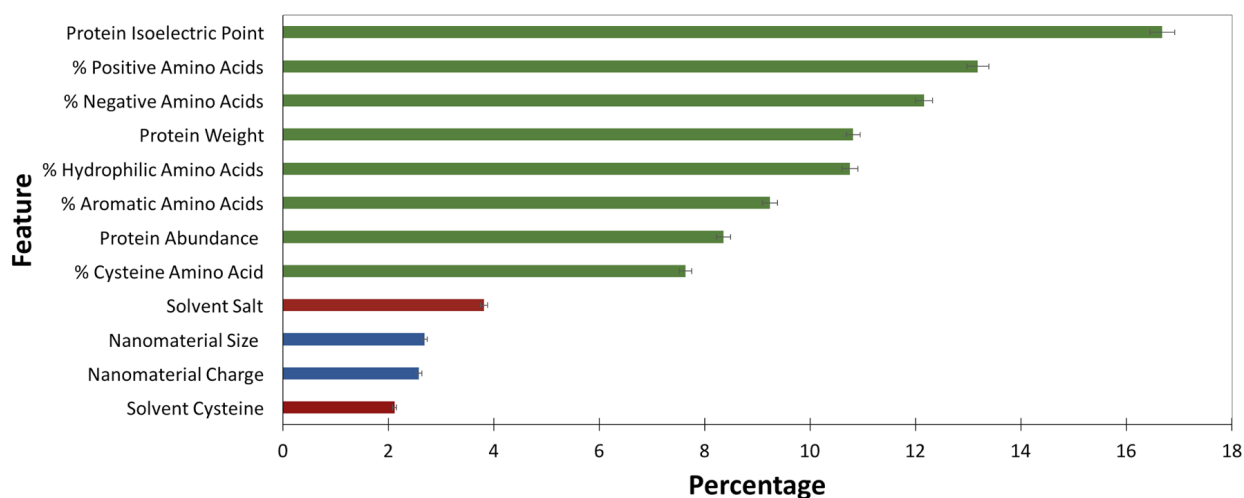


Figure 10: Weighted importance of each feature included in the final model. Protein features are shown in green, ENM features in blue, and solvent features in red. Error bars are shown with black lines.

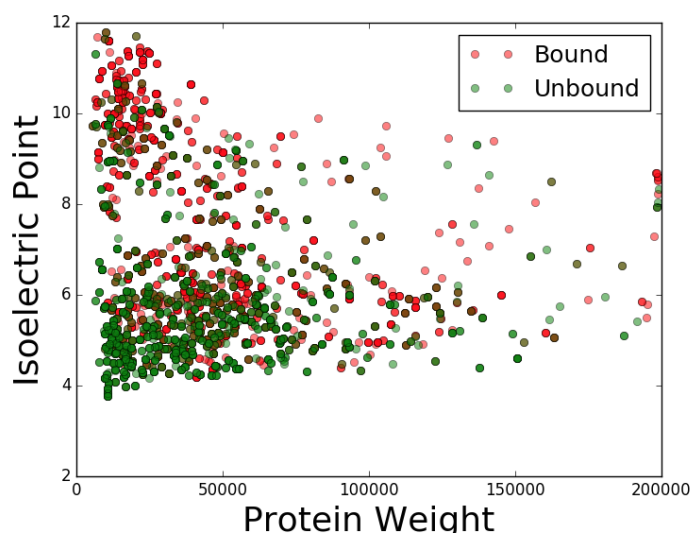


Figure 11: Clustered pI and protein weight values over the entire database. Light proteins with a high pI tend to bind to ENMs, while light proteins with lower pI tend not to bind to ENMs.

4.0 Conclusion and Future Efforts

A machine learning model was developed that predicts the PC population using protein biophysicochemical characteristics, basic ENM properties, and solution conditions. The model was proven robust with a strong AUROC (0.83), Youden index (0.50) evaluation, and has demonstrated high precision (0.77) and recall (0.85) in multiple test datasets (50). Mechanistic models of protein-ENM interactions have yet to be developed for ENMs within a complex mixture of proteins. A key feature of the machine learning method is the ability to provide a weighted list of feature importance in the model, and suggest factors mediating protein and ENM charge are most important, followed by secondary features such as protein and ENM size.

The results demonstrate that an applied machine learning approach (RFC) can enable prediction of a PC population with routine experimental data and easily accessed protein biophysical

characteristics; importantly, the model has proven robust without mechanistic insights or experimentally complex variables such as protein-protein interaction maps. As publicly available databases with quantitative protein enrichment data expand, the model can be readily tested on PC populations in other systems. Indeed, application to new datasets will enhance insights into the contribution of ENM and solvent properties in the model.

5.0 Bibliography:

- (1) Vance, M. E., Kuiken, T., Vejerano, E. P., McGinnis, S. P., Hochella, M. F., Jr., Rejeski, D. and Hull, M. S. (2015) Nanotechnology in the real world: Redeveloping the nanomaterial consumer products inventory. *Beilstein Journal of Nanotechnology*, 6, 1769-1780
- (2) Walkey, Carl D., et al. "Protein corona fingerprinting predicts the cellular interaction of gold and silver nanoparticles." *ACS nano* 8.3 (2014): 2439-2455.
- (3) Fisher, Nicholas S., and Sharon E. Hook. "Toxicology Tests with Aquatic Animals Need to Consider the Trophic Transfer of Metals." *Toxicology* 181-182 (2002): 531-36.
- (4) Judy, Jonathan D., Jason M. Unrine, and Paul M. Bertsch. "Evidence for Biomagnification of Gold Nanoparticles within a Terrestrial Food Chain." *Environmental Science & Technology* 45.2 (2011): 776-81.
- (5) Tourinho, Paula S., Cornelis A. M. Van Gestel, Stephen Lofts, Claus Svendsen, Amadeu M. V. M. Soares, and Susana Loureiro. "Metal-based Nanoparticles in Soil: Fate, Behavior, and Effects on Soil Invertebrates." *Environmental Toxicology and Chemistry* 31.8 (2012): 1679-692.
- (6) Monopoli MP, Aberg C, Salvati A, Dawson K. A. Rapid formation of plasma protein corona critically affects nanoparticle pathophysiology. *Natural Nanotechnol.* 2012;7(12):779.
- (7) Docter D, Westmeier D, Markiewicz M, Stolte S, Knauer SK, Stauber RH. The nanoparticle biomolecule corona: lessons learned – challenge accepted? *Chem Soc Rev.* 2015;44(17):6094.
- (8) Hellstrand E, Lynch I, Andersson A, Drakenberg T, Dahlback B, Dawson KA, et al. Complete high-density lipoproteins in nanoparticle corona. *FEBS J.* 2009;276:3372.
- (9) Jason R McCarthy and Farouc A Jaffer. The role of nanomedicine in the imaging and therapy

of thrombosis *Nanomedicine (Lond)*. 2011 Oct; 6(8): 1291–1293.doi: 10.2217/nnm.11.128

(10) Walkey CD, Chan WCW. Understanding and controlling the interaction of nanomaterials with proteins in a physiological environment. *Chem Soc Rev*. 2012;41(7):2780.

(11) Setyawati MI, Tay CY, Docter D, Stauber RH, Leong DT. Understanding and exploiting nanoparticles' intimacy with the blood vessel and blood. *Chem Soc Rev*. 2015;44(22):8174.

(12) Walczyk D, Bombelli FB, Monopoli MP, Lynch I, Dawson KA. What the cell “sees” in bionanoscience. *J Am Chem Soc*. 2010;132:5761.

(13) Sarhan, OM., and RM Hussein. “Effects of intraperitoneally injected silver nanoparticles on histological structures and blood parameters in the albino rat.” *International Journal of Nanomedicine* (2014). 1505-17.

(14) Abdelhalim, Mohamed Anwar K., and Sherif A. Abdelmottaleb Moussa. "The Gold Nanoparticle Size and Exposure Duration Effect on the Liver and Kidney Function of Rats: In Vivo." *Saudi Journal of Biological Sciences* 20.2 (2013): 177-81.

(15) Goldberg, E., M.; Scheringer, T. D. Buchelic, and K. Hungerbühler, Prediction of nanoparticle transport behavior from physicochemical properties: Machine learning provides insights to guide the next generation of transport models, *Environ. Sci.: Nano*, 2015, **2**, 352–360, doi:[10.1039/c5en00050e](https://doi.org/10.1039/c5en00050e).

(16) Etheridge, Michael L., et al. "The big picture on nanomedicine: the state of investigational and approved nanomedicine products." *Nanomedicine: nanotechnology, biology and medicine* 9.1 (2013): 1-14.

(17) Duran, Nelson, and Priscyla D. Marcato. "Nanobiotechnology perspectives. Role of nanotechnology in the food industry: a review." *International Journal of Food Science & Technology* 48.6 (2013): 1127-1134.

(18) Setyawati MI, Tay CY, Docter D, Stauber RH, Leong DT. Understanding and exploiting nanoparticles' intimacy with the blood vessel and blood. *Chem Soc Rev*. 2015;44(22):8174.

- (19) Eigenheer, Richard, et al. "Silver nanoparticle protein corona composition compared across engineered particle properties and environmentally relevant reaction conditions." *Environmental Science: Nano* 1.3 (2014): 238-247.
- (20) *Office of Budget*. National Institutes of Health, 2016. Web. 1 Dec. 2016.
- (21) Dahle, J.T.; Arai, Y. Environmental Geochemistry of Cerium: Applications and Toxicology of Cerium Oxide Nanoparticles. *Int. J. Environ. Res. Public Health* 2015, 12, 1253-1278.
- (22) U. Consortium, *Nucleic Acids Res.*, 2017, **45**, D158–D169.
- (23) R. D. Finn, T. K. Attwood, P. C. Babbitt, A. Bateman, P. Bork, A. J. Bridge, H.-Y. Chang, Z. Dosztányi, S. El-Gebali, M. Fraser, J. Gough, D. Haft, G. L. Holliday, H. Huang, X. Huang, I. Letunic, R. Lopez, S. Lu, A. Marchler-Bauer, H. Mi, J. Mistry, D. A. Natale, M. Necci, G. Nuka, C. A. Orengo, Y. Park, S. Pesseat, D. Piovesan, S. C. Potter, N. D. Rawlings, N. Redaschi, L. Richardson, C. Rivoire, A. Sangrador-Vegas, C. Sigrist, I. Sillitoe, B. Smithers, S. Squizzato, G. Sutton, N. Thanki, P. D. Thomas, S. C. E. Tosatto, C. H. Wu, I. Xenarios, L.-S. Yeh, S.-Y. Young and A. L. Mitchell, *Nucleic Acids Res.*, 2017, **45**, D190–D199.
- (24) A. L. Swan, A. Mobasher, D. Allaway, S. Liddell and J. Bacardit, *OMICS*, 2013, **17**, 595–610.
- (25) E. Goldberg, M. Scherlinger, T. D. Bucheli, K. Hungerbühler, C.-W. Lam, D. B. Warheit, A. B. Santamaria, M. J. McLaughlin, J. R. Lead, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *Environ. Sci. Nano*, 2015, **2**, 352–360.
- (26) L. Breiman, *Mach. Learn.*, 2001, **45**, 5–32.
- (27) D. M. W., *J. Mach. Learn. Technol. ISSN*, 2011, **2**, 2229–3981.
- (28) J. Davis and M. Goadrich, in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, ACM Press, New York, New York, USA, 2006, pp. 233–240.
- (29) M. H. Zweig and G. Campbell, *Clin. Chem.* 1993, **39**, 561-577.
- (30) P. Xu, X. Liu, D. Hadley 1#, S. Huang, J. Krischer, C. Beam, P. Xu, X. Liu, H. D. Huang and S. Krischer, *J Proteomics Bioinform*, 2014, **S9**, 006. DOI:10.4172/jpb.S9-006.

- (31) Z. R. Yang, *Brief. Bioinform.*, 2004, **5**, 328–38.
- (32) E. F. Schisterman, N. J. Perkins, A. Liu and H. Bondell, *Epidemiology*, 2005, **16**, 73–81.
- (33) B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich and F. A. Hamprecht, *BMC Bioinformatics*, 2009, 10, 213. DOI:10.1186/1471-2105-10-213.
- (34) B. H. Rihn and O. Joubert, *ACS Nano*, 2015, **9**, 5634–5635.
- (35) N. Sain, G. Tiwari and D. Mohanty, *Sci. Rep.*, 2016, 6, 31418.

Appendix A: Source Code

"""

Matthew Findlay
Santa Clara University
Dr. Wheeler's Lab

This Script predicts if proteins will be found in the protein corona on the surface of Engineered Nanomaterials.

To achieve this we first experimentally isolate proteins that bind and do not bind to engineered nanomaterials

under a variety of relevant biological conditions. We send these protein samples to Stanford's to LC-MS/MS facilities

to identify the proteins and their associated spectral counts. We then mine online databases to create a database

containing information about the proteins, particles, and solvent conditions.

To make predictions from our database we use a random forest classification algorithm.

We validate our classifications with several statistical methods including ROC curves.

"""

```
import math
import numpy as np
import pandas as pd
import sklearn
from sklearn.metrics import roc_auc_score, roc_curve, auc
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import RFECV
from sklearn.grid_search import GridSearchCV
import matplotlib.pyplot as plt
from sys import argv
import sys
import json

class RandomForestClassifierWithCoef(RandomForestClassifier):
    """Adds feature weights for each returned variable"""
    def fit(self, *args, **kwargs):
        """Overloaded fit method to include the feature importances
        of each variable. This is used for RFECV
        """
        super(RandomForestClassifierWithCoef, self).fit(*args, **kwargs)
        self.coef_ = self.feature_importances_

class validation_metrics(object):
    """Several statistical tests used to validate the models predictive power
    Takes True results of the test data and the models results as parameters for constructor
    """
    def __init__(self, true_results, predicted_results):
        self.true_results = true_results
        self.predicted_results = predicted_results
```

```

def youden_index(self, plot=1):
    """Calculates the youden_index for each threshold
    Takes no arguments except option to turn off plot
    Unless Specified, outputs a plot of Youden Index vs. thresholds
    """
    youden_index_values = []
    tpr = []
    fpr = []
    fpr, tpr, thresholds = set_threshold_roc_curve(self.true_results, self.predicted_results,
    pos_label=1, drop_intermediate=True)
    print thresholds
    for i in range(0, len(thresholds)):
        YI=((tpr[i]+(1-fpr[i])-1))/(math.sqrt(2))
        youden_index_values.append(YI)

    if(plot):
        plt.figure(figsize=(12, 9))
        ax = plt.subplot(111)
        ax.spines["top"].set_visible(False)
        ax.spines["right"].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()
        plt.xticks(fontsize=19)
        plt.yticks(fontsize=19)
        plt.ylim([0.0, 0.5])
        plt.title('Optimal Accuracy Cutoff', fontsize=22)
        plt.xlabel('Classification Cutoff Threshold', fontsize=20)
        plt.ylabel('Youden Index', fontsize=20)
        plt.plot(thresholds, youden_index_values, color="#800000", linewidth=2)
        plt.show()

def roc_curve(self, plot=1):
    """Plots the Reciever Operating Characteristic Curve
    Takes no arguments aside from option to turn off plot
    Unless specified, outputs a ROC curve
    """
    roc = roc_auc_score(self.true_results, self.predicted_results)
    fpr, tpr, thresholds = set_threshold_roc_curve(self.true_results, self.predicted_results,
    pos_label=1, drop_intermediate=True)

    if(plot):
        plt.figure(figsize=(12, 9))
        ax = plt.subplot(111)
        ax.spines["top"].set_visible(False)
        ax.spines["right"].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()
        plt.xlim([0.0, 1.01])
        plt.ylim([0.0, 1.01])
        plt.xticks(fontsize=19)

```

```

plt.yticks(fontsize=19)
plt.plot(fpr, tpr, label='Area Under the Curve=%.2f' % roc, color="#800000", linewidth=2)
plt.plot([0, 1], [0, 1], 'k--', label = 'Area Under the Random Guess Curve=0.5')
plt.xlabel('1-specificity', fontsize=20)
plt.ylabel('Sensitivity', fontsize=20)
plt.title('Receiver Operating Characteristic Curve', fontsize=22)
plt.legend(loc="lower right")
plt.show()

def well_rounded_validation(self):
    """Calculates a handful of important model validation metrics. I consider this a well rounded
    validation
    Takes no arguments
    Returns a Dict containing the AUROC, Recall, Precision, F1-score, Accuracy, and
    confusion matrix from the model
    """
    classified_predictions = classify(self.predicted_results, 0.5)
    conf_matrix = sklearn.metrics.confusion_matrix(self.true_results, classified_predictions,
    labels=None)

    return {
        "AUROC" : roc_auc_score(self.true_results, self.predicted_results),
        "Recall" : sklearn.metrics.recall_score(self.true_results, classified_predictions,
    labels=None, pos_label=1, average=None, sample_weight=None)[1],
        "Precision" : sklearn.metrics.precision_score(self.true_results, classified_predictions,
    labels=None, pos_label=1, average=None, sample_weight=None)[1],
        "F1 Score" : sklearn.metrics.f1_score(self.true_results, classified_predictions),
        "Accuracy" : sklearn.metrics.accuracy_score(self.true_results, classified_predictions),
        "Confusion Matrix" : [conf_matrix[0][0], conf_matrix[0][1], conf_matrix[1][0],
    conf_matrix[1][1]]
    }

class visualize_data(object):
    """Offers an easy way to create beautiful histograms for the input data
    Takes a target value as constructor variable (enrichment values)
    """
    def __init__(self, enrichment):
        self.enrichment = enrichment
        self.target = classify(enrichment, 1.0)

    def continuous_data_distribution(self, enrichment, particle):
        """This function creates a dank histogram of given data
        Takes a title and enrichment values as parameters
        outputs aesthetic graph
        """
        plt.figure(figsize=(12, 9))
        ax = plt.subplot(111)
        ax.spines["top"].set_visible(False)
        ax.spines["right"].set_visible(False)
        ax.get_xaxis().tick_bottom()

```



```

ax.get_yaxis().tick_left()
plt.xticks(fontsize=26)
plt.yticks(fontsize=26)
plt.ylim([0.0, 100])
plt.xlim([-3.0,3.0])
plt.hist(np.log10(enrichment), bins=25, color = "#3F5D7D")
#plt.title('Histogram of ' + str(particle), y=1.08, fontsize=22)
plt.ylabel('Frequency', fontsize=26)
plt.xlabel('Logarithmic Enrichment Factor', fontsize=26)
plt.tight_layout()
plt.show()

```

```
def visualize_by_particle(self):
```

```

    """Visualizes all the particle types in the dataset
    Takes no arguments
    Outputs 7 graphs, one for each reaction condition
    """

```

```

        self.continuous_data_distribution(self.enrichment, 'Enrichment Factors on All Particles in
The Database with 50 bins')
        self.continuous_data_distribution(self.enrichment[0:356], 'Enrichment Factors on the
Positive 10nm Silver Nanoparticle \n with no Solute')
        self.continuous_data_distribution(self.enrichment[356:924], 'Enrichment Factors on the
Negative 10nm Silver Nanoparticle \n with no Solute')
        self.continuous_data_distribution(self.enrichment[924:1502], 'Enrichment Factors on the
Negative 100nm Silver Nanoparticle \n with no Solute')
        self.continuous_data_distribution(self.enrichment[1502:1989], 'Enrichment Factors on the
Negative 10nm Silver Nanoparticle \n with 0.1mM Cysteine')
        self.continuous_data_distribution(self.enrichment[1989:2499], 'Enrichment Factors on the
Negative 10nm Silver Nanoparticle \n with 0.8 mM NaCl' )
        self.continuous_data_distribution(self.enrichment[2499:3013], 'Enrichment Factors on the
Negative 10nm Silver Nanoparticle \n with 3.0 mM NaCl')
        self.discrete_data_distribution()

```

```
def scatterplot(self, data, x, y):
```

```

    """
    Takes in the dataframe and two columns of choice.
    Outputs a 2-d scatter plot of the data
    """

```

```

        bound_x = []
        bound_y = []
        unbound_x = []
        unbound_y = []
        for i, k in enumerate(self.target):
            if k == 0:
                bound_x.append(data[x][i])
                bound_y.append(data[y][i])
            else:
                unbound_x.append(data[x][i])
                unbound_y.append(data[y][i])

```

```

line = plt.figure()

plt.plot(unbound_y, unbound_x, "o", color='r', alpha=0.5)
plt.plot(bound_y, bound_x, "o", color='g', alpha=0.5)
plt.ylim([0, max(data[x])])
plt.xlim([0, max(data[y])])
plt.legend(('Bound', 'Unbound'), fontsize=18)
plt.ylabel(str(x), fontsize = 26)
plt.xlabel(str(y), fontsize=26)

def random_number():
    """This function imports the current time in nanoseconds to use as a pseudo-random number
    Takes no arguments
    Returns pseudo-random number
    """
    from datetime import datetime
    dt = datetime.now()
    rnum = dt.microsecond
    return rnum

def optimize(model, training_data, training_results):
    """This function optimizes the machine learning classifier, returning the parameters that give
    the best accuracy
    Takes the model, training data and training targets as arguments
    Outputs the best Parameters
    """
    #add whatever your heart desires to param grid, keep in mind its an incredibly inefficient
    algorithm
    param_grid = {
        'n_estimators': [1000],
        'max_features': ['auto'],
        'max_depth': [None],
        'min_samples_split': [5],
        'min_samples_leaf': [1],
        'n_jobs': [-1],
        'random_state' : [46, 0]
    }
    #5 fold validation
    CV_est = GridSearchCV(estimator=model, param_grid=param_grid, cv= 5)
    CV_est.fit(training_data, training_results)
    print CV_est.best_params_

def recursive_feature_elimination(model, training_data, training_results):
    """Runs RFECV with 5 folds
    Takes model, training features, and targets as command line arguments
    Outputs optimum features
    """
    selector = RFECV(estimator=model, step=1, cv=5, scoring='roc_auc', verbose=1)
    selector = selector.fit(training_data, training_results)
    print selector.support_

```

```

print "\n"
print selector.ranking_
print "\n"
print "Optimal number of features: "
print selector.n_features_
print "\n"
print selector.grid_scores_

def get_dummies(dataframe, category):
    """This function converts categorical variables into dummy variables
    Takes pandas dataframe and the category name as arguments
    Returns the dataframe with new dummy variables
    """
    dummy = pd.get_dummies(dataframe[category], prefix=category)
    dataframe = pd.concat([dataframe,dummy], axis = 1)
    dataframe.drop(category, axis=1, inplace=True)
    return dataframe

def classify(proba, cutoff):
    """This function classifies particles as bound or unbound
    Takes unclassified data the cutoff as arguments
    returns classified data in a list
    """
    predicted_results = []
    for i in proba:
        if i >= cutoff:
            temp = 1
        else:
            temp = 0
        predicted_results.append(temp)

    return predicted_results

def stable_cumsum(arr, rtol=1e-05, atol=1e-08):
    """
    Taken From sci-kit learn documentation to help set_threshold_roc_curve
    Altered to give a constant amount of thresholds for the roc curve
    """
    out = np.cumsum(arr, dtype=np.float64)
    expected = np.sum(arr, dtype=np.float64)
    if not np.allclose(out[-1], expected, rtol=rtol, atol=atol):
        raise RuntimeError('cumsum was found to be unstable: '
                           'its last element does not correspond to sum')
    return out

def set_threshold_roc_curve(y_true, y_score, pos_label=None, sample_weight=None,
                           drop_intermediate=True):
    """
    Taken from sci-kit learn documentation
    Altered to give a constant amount of thresholds for the roc curve

```

```

"""
fps, tps, thresholds = _binary_clf_curve(
    y_true, y_score, pos_label=pos_label, sample_weight=sample_weight)
if tps.size == 0 or fps[0] != 0:
    # Add an extra threshold position if necessary
    tps = np.r_[0, tps]
    fps = np.r_[0, fps]
    thresholds = np.r_[thresholds[0] + 1, thresholds]
if fps[-1] <= 0:
    warnings.warn("No negative samples in y_true, "
                  "false positive value should be meaningless",
                  UndefinedMetricWarning)
    fpr = np.repeat(np.nan, fps.shape)
else:
    fpr = fps / fps[-1]
if tps[-1] <= 0:
    warnings.warn("No positive samples in y_true, "
                  "true positive value should be meaningless",
                  UndefinedMetricWarning)
    tpr = np.repeat(np.nan, tps.shape)
else:
    tpr = tps / tps[-1]
return fpr, tpr, thresholds

def _binary_clf_curve(y_true, y_score, pos_label=None, sample_weight=None):
    """
    Taken from sci-kit learn documentation to help set_threshold_roc_curve()
    Altered to return a constant amount of thresholds for each roc curve
    Calculate true and false positives per binary classification threshold.
    """
    if sample_weight is not None:
        sample_weight = column_or_1d(sample_weight)
    # ensure binary classification if pos_label is not specified
    classes = np.unique(y_true)
    if (pos_label is None and
        not (array_equal(classes, [0, 1]) or
            array_equal(classes, [-1, 1]) or
            array_equal(classes, [0]) or
            array_equal(classes, [-1]) or
            array_equal(classes, [1]))):
        raise ValueError("Data is not binary and pos_label is not specified")
    elif pos_label is None:
        pos_label = 1.
    # make y_true a boolean vector
    y_true = (y_true == pos_label)
    # sort scores and corresponding truth values
    desc_score_indices = np.argsort(y_score, kind="mergesort")[::-1]
    y_score = y_score[desc_score_indices]
    y_true = y_true[desc_score_indices]
    if sample_weight is not None:

```

```

        weight = sample_weight[desc_score_indices]
    else:
        weight = 1.
    # y_score typically has many tied values. Here we extract
    # the indices associated with the distinct values. We also
    # concatenate a value for the end of the curve.
    distinct_value_indices = np.where(np.diff(y_score))[0]
    threshold_idxs = np.r_[distinct_value_indices, y_true.size - 1]
    # accumulate the true positives with decreasing threshold
    tps = stable_cumsum(y_true * weight)[threshold_idxs]
    if sample_weight is not None:
        fps = stable_cumsum(weight)[threshold_idxs] - tps
    else:
        fps = 1 + threshold_idxs - tps
    return fps, tps, y_score[threshold_idxs]

def clean_print(obj):
    """
    Prints the JSON in a clean format for all my
    Biochemistry friends
    """
    if type(obj) == dict:
        for key, val in obj.items():
            if hasattr(val, '__iter__'):
                print "\n" + key
                clean_print(val)
            else:
                print '%s : %s' % (key, val)
    elif type(obj) == list:
        for val in obj:
            if hasattr(val, '__iter__'):
                clean_print(val)
            else:
                print val
    else:
        print str(obj) + "\n"

def fetch_data():
    """
    Pulls the Data from CSV format. Returns 3012 measured protein-particle
    interactions represented as vectors
    """
    try:
        data = pd.read_csv("train.csv")
        target = pd.read_csv("class_result.csv")
        enrichment = pd.read_csv("result.csv")
    except:
        "Error Fetching CSV Data"
    #One hot encoding of categorical data
    data = get_dummies(data, 'size')

```

```

data = get_dummies(data, 'charge')
data = get_dummies(data, 'salt')
data = get_dummies(data, 'cysteine')
#Fill NaN's with average value in Abundance data
count = 0
total = 0
for val in data['Abundance']:
    if not np.isnan(val):
        count+=1
        total+=val
data = data.fillna(total/count)
#Normalize the data
min_max_scaler = sklearn.preprocessing.MinMaxScaler()
np_scaled = min_max_scaler.fit_transform(data)
df_normalized = pd.DataFrame(np_scaled)
#Classify enrichment data, using enrichment ratio of 1
classed_enrich = []
for i in enrichment.itertuples():
    if i[1] >= 1:
        temp = 1
    else:
        temp = 0
    classed_enrich.append(temp)
#split data into training and testing set. Use testing set to validate model at the end
training_data, test_data, training_results, test_results =
sklearn.cross_validation.train_test_split(df_normalized, classed_enrich, test_size=0.1,
random_state = random_number())
training_results= np.ravel(training_results)
#Ravel those vectors
test_results = np.ravel(test_results)
enrichment = np.ravel(enrichment)
target = np.ravel(target)

return training_data, test_data, training_results, test_results, enrichment, target, data

def main():
    training_data, test_data, training_results, test_results, enrichment, target, data = fetch_data()
    #Visualize the data
    vis = visualize_data(enrichment)
    vis.visualize_by_particle()
    #vis.scatterplot(data, 'Pi', 'Weight')
    #Print Relevant information
    print "Amount of Training data: " + str(len(training_data))
    print "Amount of Testing Data: " + str(len(test_data))

    est = RandomForestClassifierWithCoef(
        #criterion='mse',          #mean squared error criterion
        n_estimators=10000,        #number of trees used by the algorithm
        oob_score=True,           #Out of box score
        max_features='auto',      #features at each split (auto=all)

```

```

        max_depth=None,          #max tree depth
        min_samples_split=5,      #minimum amount of samples to split a node
        min_samples_leaf=1,       #minimum amount of samples a leaf can
contain
        min_weight_fraction_leaf=0, #minimum weight fraction of samples in a
leaf
        max_leaf_nodes=None,      #maximum amount of leaf nodes
        n_jobs=-1,                #CPU Cores used (-1 uses all)
        random_state=random_number() #Initialize random seed generator
    )

    est.fit(training_data, training_results)          #fit model to training data
    #Get prediction probabilities
    probability_prediction = est.predict_proba(test_data)[: ,1]
    #Feature importance based on entropy calculations
    features = dict(zip(list(data), est.feature_importances_))
    #Run validation Metrics
    #val = validation_metrics(test_results, probability_prediction)
    #val.roc_curve()
    #val.youden_index()
    return val.well_rounded_validation(), features

if __name__ == '__main__':
    results = {}
    for i in range(0, int(argv[1])):
        metrics = main()
        results["Run_" + str(i)] = metrics[0], metrics[1]
    print json.dumps(results)

```

Appendix B: Undergraduate Funding Request Letter

Team Information:

Name	University Affiliation	Department	Primary Contact	e-mail
Matthew Findlay	Student	Bioengineering	Yes	mfindlay@scu.edu
Daniel Freitas	Student	Bioengineering	No	dnfreitas@scu.edu
Maryam Mobed-Miremadi	Professor	Bioengineering	No	mmobedmiremadi@scu.edu
Korin Wheeler	Professor	Chemistry & Biochemistry	No	kwheeler@scu.edu

Abstract: The use of engineered nanoparticles (ENMs) in consumer and commercial products is increasing rapidly (Figure 1). The small size and high surface reactivity of ENMs allows them to be easily incorporated into various materials, and gives them a range of properties including but not limited to bacterial resistance, semi-conductance, and UV ray resistance. These properties make ENMs very appealing to modern industry, but also make ENMs toxic, causing serious health and environmental concerns. As a result of the increasing release of ENMs into the global ecosystem, our team aims to ensure that ENMs are designed to minimize damage to the global ecosystem and human health. Protein absorption creates a surface corona on ENMs and plays a large role in ENM toxicity¹. Despite the role of the protein corona in driving ENM behavior, the field is far from a model that offers predictive insight toward elucidating the protein corona composition of ENMs under the broad spectrum of relevant biological conditions. Using LC-MS/MS proteomics previously established procedures, the proteins enriched on the surface of silver ENMs can be characterized². Specifically, a matrix of relevant protein-ENM reactions will be evaluated, including a range of particle sizes, surface coatings, and environmental conditions (e.g. salt concentrations). Machine learning (random forest) regression and classification will be applied to this protein enrichment data matrix to evaluate the competing roles of the biophysical properties of proteins, ENM properties themselves, and solution conditions, in mediating the formation of the ENM protein corona. The resulting model will aim to offer an accurate prediction of protein enrichment on the surface of ENMs across complex biological systems using the random forest classifier method. This statistical framework offers a crucial step forward in solving design concerns with engineering nanomaterials to minimize unintended interactions with biological and environmental systems while maximizing the efficiency of the ENMs engineered purpose.

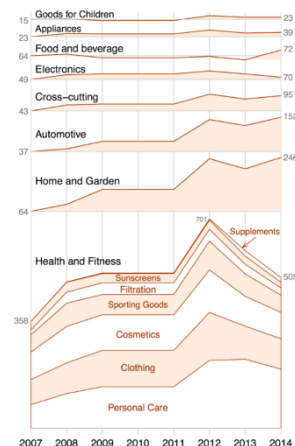


Figure 1. This graphic shows the number of products (y-axis) that have incorporated ENMs from 2007-14 (From citation 3).

Ethical Ramifications: As the usage of ENMs by modern industry increases, harmful ENMs are released into our global ecosystem (Figure 2), when ENMs are manufactured and consumed, ENM waste is released into the environment. This waste compounds over time, and has an impact on our global ecosystem by entering waterways, soil, food, residential areas, and eventually humans. This range of ecological impacts motivates our labs to offer the framework to design products that are not only safer to humans, but to the environment that they contaminate. Nanotechnology is still a young field, but it is growing rapidly. Nanotechnology has the potential to mold the future by creating vast technological and medical advancements. While our labs understand that devastating, large-scale abuse of the global environment occurs on a daily basis, we stress that these large scale issues cause society to overlook small-scale

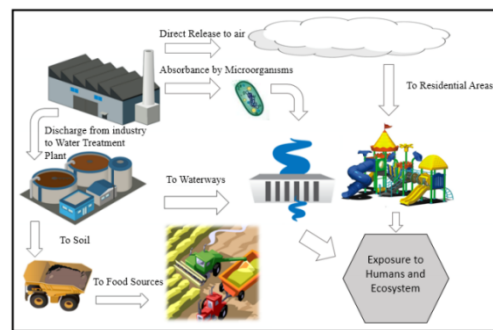


Figure 2: A schematic of how ENMs are spread from industry into the ecosystem, impacting human and environmental health. (Modified from citation 4)

pollutants, such as ENMs, that do tangible damage to our global ecosystem. The nanotechnology market is expected to grow at an exponential rate, and has the potential to revolutionize modern industry. It is important that steps toward safer products are taken while the nanomaterial industry is still young, as these steps will pave the way for the nanomaterial industry to grow in an ethical manner. Today, it is apparent that industry growth is often seen as more important than environmental, worker, and human safety, this is why it is crucial that the proper scientific research is conducted to educate the public and give environmental and human health the attention it deserves.

Internal Funding Sources:

Willem P. Roelandts and Maria Constantino-Roelandts grant (**\$3,000**)

Budget:

Experiments	Cost
(-) Silver 10nm ENM proteomics experiment at Stanford using LC-MS/MS	\$900
(+) Silver 10nm ENM proteomics experiment at Stanford using LC-MS/MS	\$900
(-) Silver 100nm ENM proteomics experiment at Stanford using LC-MS/MS	\$900
(+) Silver 10nm ENM with 3.0mM NaCl proteomics experiment at Stanford using LC-MS/MS	\$900
(-) Silver 100nm ENM with 3.0mM NaCl proteomics experiment at Stanford using LC-MS/MS	\$900
(-) Silver 10nm ENM with 3.0mM NaCl proteomics experiment at Stanford using LC-MS/MS	\$900
Total:	\$5,400

Undergraduate Programs Funding:

We are asking Undergraduate Programs to fund us up to **\$2,000**. This in combination with the funding we have from the Willem P. Roelandts and Maria Constantino-Roelandts grant will pay the **\$5,400** necessary to complete our experimentation. Additional minor experimental costs necessary for publication of this work (e.g. TEM characterization of the nanomaterials) will be paid for by Prof. Korin Wheeler's NIH AREA grant (Grant Number: 1 R15 ES025929-01A1); since MS proteomics is beyond the scope of the NIH funding, additional funds are necessary to cover these costs.

Xilinx Funding:

We are also looking for additional funding support from Xilinx. Addition support will allow our team to expand the scope of our project by giving us the opportunity to evaluate the proteins on the surface of uncharged ENMs. This will increase the robustness of our statistical model and offer deeper insight into the role of protein absorption on ENM toxicity. The ethical ramifications of showing protein affinity to neutral ENMs is significant and can lead to reduced environmental and human toxicity. We are requesting **\$1,800** to run experiments on 10nm and 100nm uncharged ENMs.

Experiments	Cost
Neutral Silver 10nm ENM proteomics experiment at Stanford using LC-MS/MS	\$900
Neutral Silver 100nm ENM proteomics experiment at Stanford using LC-MS/MS	\$900
Total:	\$1,800

References:

- (1) Fisher, Nicholas S., and Sharon E. Hook. "Toxicology Tests with Aquatic Animals Need to Consider the Trophic Transfer of Metals." *Toxicology* 181-182 (2002): 531-36.
- (2) Eigenheer, Richard, Erick R. Castellanos, Meagan Y. Nakamoto, Kyle T. Gerner, Alyssa M. Lampe, and Korin E. Wheeler. "Silver Nanoparticle Protein Corona Composition Compared across Engineered Particle Properties and Environmentally Relevant Reaction Conditions." *Environmental Science: Nano Environ. Sci.: Nano* 1.3 (2014): 238
- (3) Vance, M. E., Kuiken, T., Vejerano, E. P., McGinnis, S. P., Hochella, M. F., Jr., Rejeski, D. and Hull, M. S. (2015) Nanotechnology in the real world: Redeveloping the nanomaterial consumer products inventory. *Beilstein Journal of Nanotechnology*, 6, 1769-1780
- (4) Dahle, J.T.; Arai, Y. Environmental Geochemistry of Cerium: Applications and Toxicology of Cerium Oxide Nanoparticles. *Int. J. Environ. Res. Public Health* 2015, 12, 1253-1278.