Predicting Drug-Drug Interaction Severity Using Network Characteristics

Nagri, Deetya

Abstract

Drug-drug interactions (DDIs), which can add to or diminish the effect of one drug or impact the metabolism of one drug, have harmful effects on health in patients that take multiple drugs. Testing for DDIs is slow and costly, so computational models have recently been used to predict them. Network information is useful in describing a drug's known interactions and mechanisms to determine whether two drugs could be interacting. This research explores various machine learning models to predict the severity of unknown interactions of existing drugs (major, minor, or moderate), using the DDInter database. The best-performing model, a multi-layer perceptron model which concatenates two drugs' embeddings, had an accuracy of about 92%, average precision of about 87.1% (89.4% for the major category, 78.5% for minor, and 93.5% for moderate) and average recall of 83.3% (86.4% for major, 70.1% for minor, and 95.3% for moderate). Future work in this area includes refining models to increase precision and recall, obtaining a more balanced dataset, predicting the mode of interaction, and using other drug characteristics. The code for this project can be found here, along with a Streamlit deployment of the model: https://github.com/deetyabn/DDI_severity_prediction.

Introduction

Adverse drug-drug interactions (DDIs) are a serious threat to many people, especially the elderly who often take multiple drugs to treat various conditions. By causing 26% of adverse drug events each year, DDIs can be dangerous by causing injury or death while increasing burdens on already overworked healthcare systems [13]. DDIs can also add to or diminish the effects of one or both drugs, further harming patients' health. Drugs are also sometimes removed from the market if they have major interactions with other common drugs, such as the antihistamine drug astemizole (network article), leading to thousands of wasted dollars in development and sale in addition to the toll on human health as a result of the dangerous or fatal interaction [3]. Drug-drug interactions have a significant impact on patient health, prescribing habits, and the healthcare system, so they should be monitored and predicted to prevent health effects and expensive development costs.

Drug-drug interactions are typically categorized based on their severity and mechanism. Most DDIs are pharmacokinetic interactions in which one drug interferes with the metabolism, distribution, absorption, or elimination of the other drug. This is dangerous since it changes the concentration of the drug in the patient's system, thereby leading to decreased or increased treatment compared to what is necessary [3, 13]. Pharmacodynamic interactions, a smaller class

of DDIs, occur when one drug adds to or diminishes the effect of one or both drugs since one interferes with the other at the target site [3, 13]. In terms of severity, DDIs are commonly classified as minor, moderate, and major. Minor DDIs are usually of little clinical significance, although they can cause minor discomfort and warrant monitoring. Moderate DDIs call for closer monitoring and dosage changes, while major DDIs can cause serious adverse effects and should be avoided [3].

Ideally, DDIs could be predicted in early stages of drug development to avoid the cost of production and, for existing drugs, before adverse effects are observed in people. Various *in silico* and *in vitro* methods have been introduced for early stage prediction of DDIs and late stage prediction often involves *in vivo* tests and clinical trials. Many, however, focus on pharmacokinetic interactions and can be very specific or less effective [3, 5]. Some DDIs go undetected into the market, leading to adverse effects, increased costs, and potential market removal. Once a drug has entered the market, compiled information about adverse events in the past can be used to identify interactions and avoid them in the future, but information is limited since few people take any given drug combination [3]. Testing for DDIs is expensive and lengthy due to the expansive list of potential combinations, so computational methods can be useful in directing drug safety professionals on where they should focus their resources and efforts.

Many previously introduced *in silico* methods for DDI prediction use drug structure or similarity to predict DDI presence [5, 16, 19, 24]. Cami et al. also previously introduced a model using network-based characteristics as well as other taxonomic and intrinsic properties of drugs to predict DDIs of a certain severity, splitting their data into networks for each severity level then creating a link-prediction model for each [3]. This research instead aims to use network data to predict the severity level of DDIs. Network data is well-suited for such biological, medical, and pharmacological data since complex relationships exist within a system between many drugs, proteins, and diseases. It is hypothesized that a drug with a known interaction will also interact with drugs that have similar network positions, so this information could be useful in predicting interactions. To use network information, a weighted network of known drug-drug interactions was created based on information in the DDInter database. The nodes represented drugs and edges represented the interaction between two drugs, with an assigned weight based on the interaction severity [23]. Node2Vec, a Python package, created embeddings for each drug's network characteristics, which were then trained into various models to predict the severity of the interaction [9]. The models were evaluated based on a portion of the known data not used in training. Drug safety professionals can then use this model and other existing computational models as a guide to focus their efforts on drug pairs predicted to have major or moderate interactions.

Materials and Methods

Dataset

The DDInter database was used to train and test the models observed. This database tracks the presence and severity of DDIs,which lists known interactions between two drugs and the severity of that interaction. The dataset can be accessed here as the "ddinter_downloads_code_L.csv" download: http://ddinter.scbdd.com/download/ [23]. This subset pertains to interactions between antineoplastic drugs, which are used to treat cancer, and immunomodulating agents, which stimulate or suppress the immune system. Each row of the dataset contains 2 drugs, their ID codes in the DDInter database, and the severity of their interaction: minor, moderate, major, or unknown. The curated subset of the database consists of 65389 examples describing interactions between 1699 unique drugs. **Figure 1** shows how many interactions of each severity level are contained in the dataset.

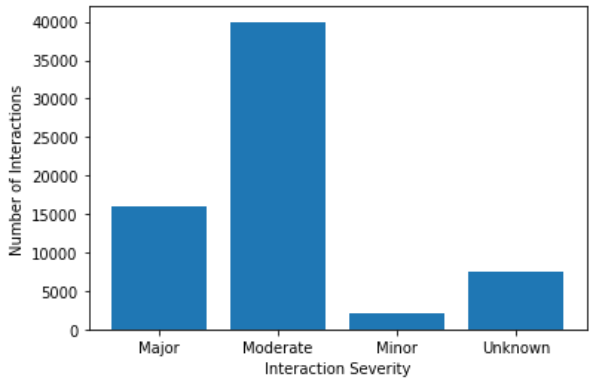

Figure 1: Distribution of DDI Severity. The data distribution favors moderate interactions over other categories.

Data Processing

The package NetworkX was used to create a weighted network using an edge-list created from the dataset [2]. Interactions with a greater severity were given greater weight when creating the network: "unknown" interactions were given a weight of 0, "minor" interactions had a weight of 1, "moderate" interactions had a weight of 2, and "major" interactions had a weight of 3. A subset of the graph with 100 edges is visualized in **Figure 2**. Many drugs had hundreds of interactions, so only a subset was pictured. **Figure 3** shows the distribution of the number of interactions each drug was involved in.

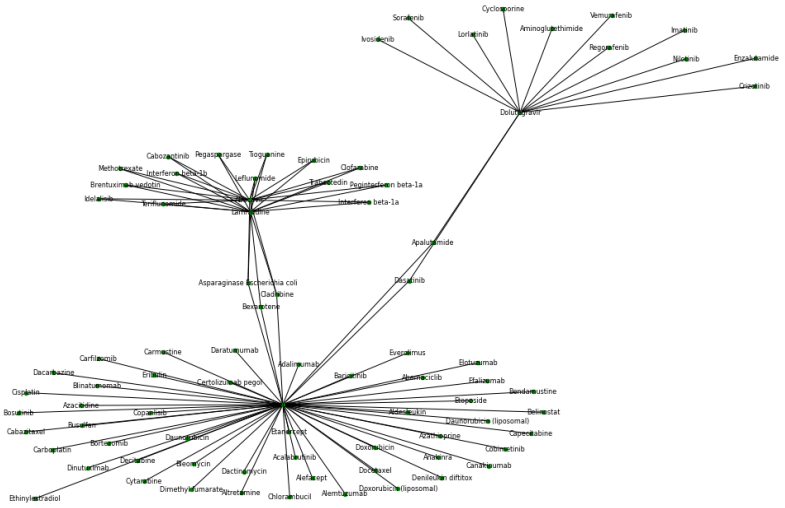

Figure 2: A Visualization of Portion of the DDI Network. This graph consists of 100 DDIs, a small portion of the entire graph that was used to create the embeddings.

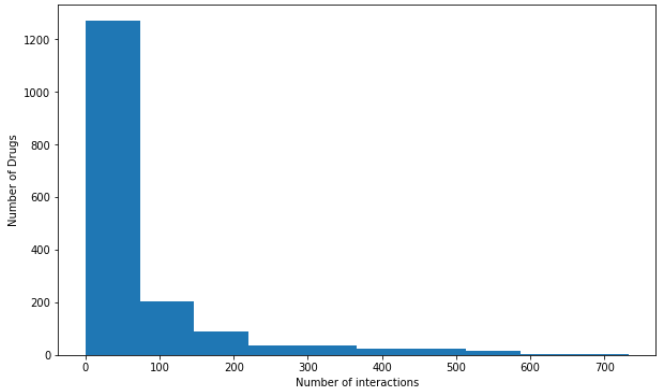

Figure 3: Distribution of Number of Interactions Each Drug is Involved In. Most drugs had between 0 and 50 interactions, but some had up to 700.

From this network, embeddings describing each drug's network characteristics were created using the package Node2Vec. Node2Vec is a Python package created by Aditya Grover and Jure Leskovec that takes graph and map nodes to an embedding space with fewer dimensions while preserving the graph's structure [9, 22]. This is done by taking random walks through the graph with each step determined based on transition probabilities, which take into account edge presence and weight [9, 22]. These sequences are then run through a Skip-Gram model like that used in Word2Vec, which is a simple neural network that inputs a word, or in this case step, sequence and outputs the probability that two are associated and their similarity [20, 22]. This yields embeddings for each node, with closer embeddings representing more similar nodes. Node2Vec was used to create embeddings for each drug with 50 dimensions based on 5 walks of length 80 nodes. **Figure 4** shows the architecture of the Node2Vec algorithm.

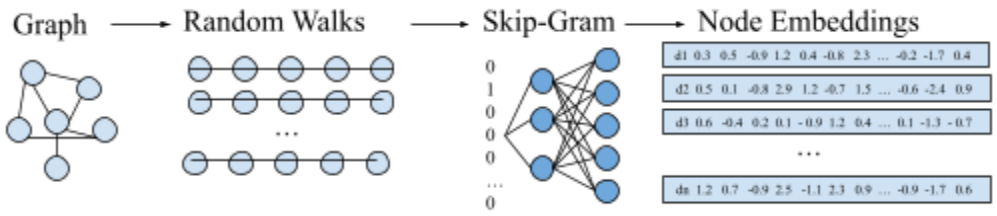

Figure 4: Node2Vec Architecture. The package Node2Vec was used to make embeddings describing the network characteristics of each drug. To do this, it uses random walks along the network graph, then feeding them into a skip-gram model to get a vector.

Only interactions of known severity were used in the final X and y matrices to feed into the model. The X matrix consisted of the embeddings of the two drugs involved in the interaction concatenated. The y matrix consisted of the severity of each interaction. These matrices were split into training and testing data, with 70% of the data used for training.

Models

Various Scikit-Learn models were tested, specifically a logistic regression model (LR), K-nearest neighbors model (KNN), decision tree model (DT), multi-layer perceptron model (MLP), and support vector machine (SVM) [17]. Logistic regression models estimate the parameters of a logistic model that is used to calculate the probability that a data point belongs to a category, making the prediction as the category with the highest probability. Logistic regression models are most commonly used in binary classification, but can also be used in multiclass classification. K-nearest neighbors models look at the k closest known data points to the unknown point and assume that the new point belongs to the same category as the majority of the near points. Decision tree models ask various questions and split the data points into categories based on their answers. A multi-layer perceptron model is a type of simple neural with one hidden layer, each node of which has a nonlinear activation function. This allows it to separate categories nonlinearly. Support vector machines, a widely used algorithm in both regression and classification models, find a hyperplane in an n-dimensional space that separates points in each category with the maximum possible distance. This model is commonly used because it has greater accuracy than logistic regression models, which work similarly, without much more computational power required.

Hyperparameters for the KNN, MLP, and SVM models were optimized with a grid search. For the KNN model, the value of k, the number of neighbors looked at in the model, was tuned. Different maximum numbers of iterations were tested for the MLP model. For the SVM model, a polynomial was used as the base function and the degree of the polynomial was varied. The maximum depth for the DT model was also varied with a random search.

The model was deployed through Google Colab and Streamlit. The code file for this is included on the Github repository for the project, where the initial code for testing the model is also included (https://github.com/deetyabn/DDI_severity_prediction).

Results and Discussion

Various performance metrics were used to test each model, including accuracy, precision, recall and a confusion matrix. The logistic regression model had approximately 73.8% accuracy, a precision of 65.1% for the major category, 73.9% for minor, and 75.3% for moderate, and recall of 34% for major, 19.6% for minor, and 92.7% for moderate. **Figure 5** shows the confusion matrix for this model. The model had trouble distinguishing between the major and moderate categories, misclassifying 3195 major interactions as moderate and having low recall for this category. 836 moderate interactions were also misclassified as major, but the size of this category prevented low recall. It also struggled with the minor category, predicting 440 minor interactions to be moderate. This is likely due to the imbalance of interaction severities in the dataset. There were significantly more moderate interactions than major or minor ones in the initial data, so this may have biased the model toward this result. Logistic regression models also tend to be most accurate for binary classification models since categories are more distinct and easily separated, which may have affected this model's metrics.

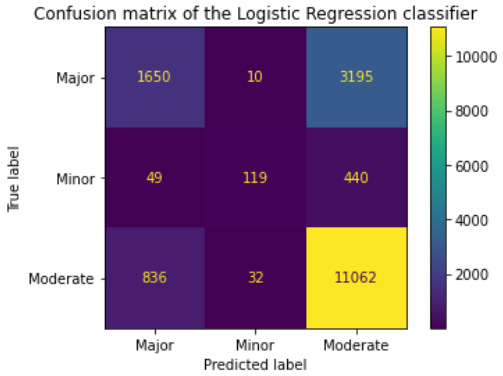

Figure 5: Confusion Matrix of the Logistic Regression Classifier. The model often switched major and moderate interactions, misclassifying them, and was heavily skewed toward predicting moderate interactions.

The initial decision tree model, which had a maximum depth of 5, had an accuracy of about 73.6%, precision of 79.5% for the major category, 83.6% for minor, and 73.1% for moderate, and recall of 22.9% for major, 7.6% for minor, and 97.6% for moderate. This tree was likely biased since the moderate class dominated, leading to very low recall for the major and minor categories. Varying the maximum depth increased the accuracy of the model so it was able to predict each category. The final decision tree model, which had a maximum depth of 41, had an accuracy of about 86.3%, precision of 79% for the major category, 61.3% for minor, and 90.6% for moderate, and recall of 79.7% for major, 63.8% for minor, and 90.1% for moderate. **Figure 6**

shows the confusion matrix for this model. This model performed much better with the major and minor interactions, though their precision and recall was still lower than that of the moderate category due to the data imbalance. The larger depth and more complex tree allowed the model to improve identification of these categories, with higher recall for minor and major with the final model than initial. However, it still switched major and moderate interactions often and misclassified 197 moderate interactions as minor and 179 minor interactions as moderate, and had low precision for the minor category, so it still had trouble distinguishing between some categories.

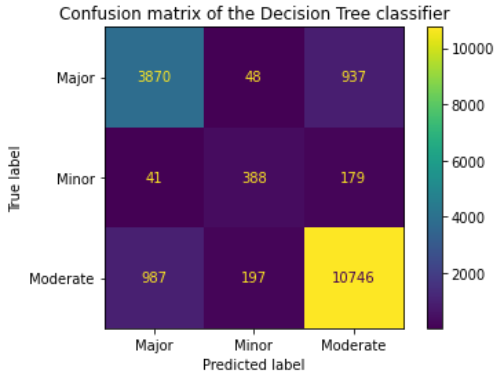

Figure 6: Confusion Matrix of the Final Decision Tree Model. The model often switched moderate and major interactions and was still skewed toward moderate predictions, but performed better with the minor interactions than the Logistic Regression Model.

The initial KNN model, which looked at 5 neighbors, had an accuracy of 83.2%, precision of 74.8% for the major category, 80.5% for minor, and 86.3% for moderate, and recall of 68.1% for major, 61.2% for minor, and 90.5% for moderate. This model performed fairly well, and was once again slightly imbalanced and had better recall for the moderate category. The final KNN model, which looked at 8 neighbors, had an accuracy of about 82.9%, precision of 72.6% for the major category, 83.5% for minor, and 86.9% for moderate, and recall of 70.7% for major, 63.3% for minor, and 88.9% for moderate. **Figure 7** shows the confusion matrix for this model. This model was more balanced in precision and recall for the three categories. This model likely performed better than the logistic regression model due to its mechanism. It is likely that two pairs with similar embeddings, which have similar network characteristics, would interact similarly and have the same severity, so this model lends itself well to the task. The model, however, did not perform as well as expected, misclassifying 1260 moderate interactions as major, 1409 major interactions as moderate, and 186 minor interactions as moderate. This is likely due to the imbalance in the data, high dimensionality of the data and the high number of samples, all of which can make a KNN model perform more poorly.

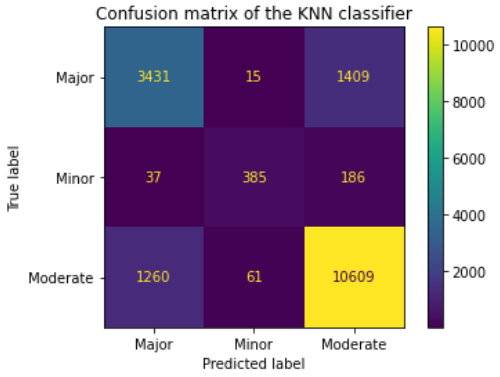

Figure 7: Confusion Matrix of the Final K-Nearest Neighbors Model. The model was more prone to switching moderate and major interactions and was still skewed toward the moderate category.

The initial SVM model, which used the default parameters for the sklearn Support Vector Classification model, had 89% accuracy. Its precision was 90.7% for the major category, 96.5% for minor, and 88.1% for moderate and its recall was 73.7% for major, 50% for minor, and 97% for moderate. This model performed better with the major category, but poorly with the minor interactions, likely due to the much smaller size of this category. The final SVM, which used a degree 4 polynomial kernel, had an accuracy around 91.6%. Its precision was 92.4% for major, 96.8% for minor, and 91.1% for moderate and its recall was 81.2% for major, 60.2% for minor, and 97.4% for moderate. **Figure 8** shows the confusion matrix for this model. This model had improved precision and recall for all the categories, including the minor category. It still, however, struggled most with the minor category, misclassifying 229 interactions as moderate likely because of the data imbalance. Like many of the other models, it also switched major and moderate interactions often, so interactions in these categories likely have similar network characteristics. The SVM model probably performed better than the KNN model because it can handle high-dimensional data and many data points more easily. Since it forms a hyperplane in n-dimensional space, it can more easily separate the 3 classes than a logistic regression or decision tree model, which are often more suited toward binary classification.

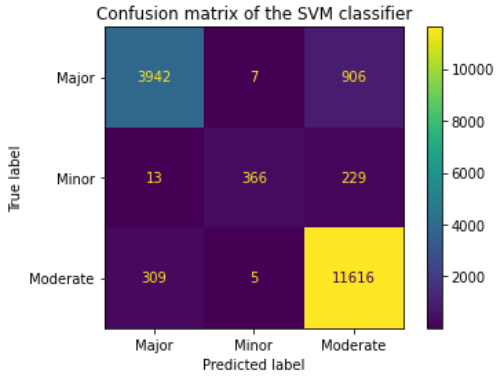

Figure 8: Confusion Matrix of the Final Support Vector Machine Model. This model was still skewed toward a moderate prediction, but was less prone to misclassifying moderate interactions as major.

The initial MLP model, which had alpha = 0.001 and 1000 maximum iterations, had accuracy of about 91.6%. Its precision was 87.6% for the major category, 73.6% for minor, and 94.1% for moderate and its recall was 88.2% for major, 68.3% for minor, and 94.2% for moderate. This model had similar weak spots as the other models, often switching major with moderate and minor with moderate interactions, leading to lower precision and recall for the minor category. The final MLP model, which had alpha=0.0001 and 200 maximum iterations, had accuracy of about 92%. Its precision was 89.4% for the major category, 78.5% for minor, and 93.5% for moderate and its recall was 86.4% for major, 70.1% for minor, and 95.3% for moderate. **Figure 9** shows the confusion matrix for this model. This model was more balanced in the precision and recall for the three categories. It misclassified 156 minor interactions as moderate, 89 moderate interactions as minor, 470 moderate interactions as major, and 630 moderate interactions as major. These misclassifications are likely due to the imbalance in the data since they involve the moderate category, which is largely overrepresented. The MLP model's more complex neural network architecture likely allowed it to learn more about the characteristics of each category and better understand how to differentiate them.

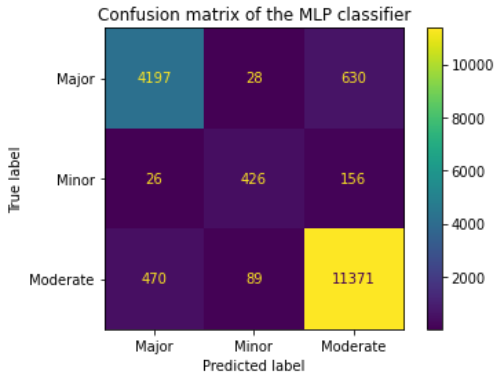

Figure 9: Confusion Matrix of the Final Multi-Layer Perceptron Model. The model was less skewed toward a moderate prediction and switched the major and moderate categories less often, but still had these limitations.

Table 1 shows a comparison of each model's accuracy, precision, and recall. For the DT, KNN, SVM, and MLP, the final, optimized model's statistics were listed. Based on this, the MLP model was chosen as the best model since it had the highest accuracy and was well-rounded.

Table 1: Accuracy, Precision, and Recall of Tested Models

Model	**Accuracy**	**Precision** ([Major, Minor, Moderate])	**Recall** ([Major, Minor, Moderate])
LR	73.8%	[65.1%, 73.9%, 75.3%]	[34%, 19.6%, 92.7%]
DT	86.3%	[79%, 61.3%, 90.6%]	[79.7%, 63.8%, 90.1%]
KNN	82.9%	[72.6%, 83.5%, 86.9%]	[70.7%, 63.3%, 88.9%]

SVM	91.6%	[92.4%, 96.8%, 91.1%]	[81.2%, 60.2%, 97.4%]
MLP	92.0%	[89.4%, 78.5%, 93.5%]	[86.4%, 70.1%, 95.3%]

This study has a number of limitations. The data used was a limited portion of the DDInter database, so it had fewer DDIs to learn from and make predictions from. The set consisted of 1,699 drugs with 65,389 interactions, but these were heavily skewed toward the moderate classification, adding bias to the models that was apparent in their performance. There is also no "gold standard" for classifying DDIs, so all severity classifications were based on the database's criteria, which may differ from other doctors' or pharmacists'. Various databases disagree about the existence and severity of drug-drug interactions, so even the correctly classified interactions may have more or less severe effects than its classification implies. Some pairs predicted to have major or moderate severity may also be rarely or never prescribed together, decreasing the use of the model.

Conclusions

Using the package Node2Vec, embeddings describing the network characteristics of 1699 unique drugs were successfully created based on a subset of the DDInter database. By testing various models from the Python package Scikit-learn on this processed dataset, a model that can predict the severity of drug-drug interactions based on each drug's network characteristics has been produced. The final model using a multi-layer perceptron model, a simple neural network, had 92% accuracy. The initial dataset used was unbalanced, with many more moderate interactions than the other classes, biasing the model toward that prediction and affecting its accuracy.

Future work in this topic could include building a similar model with a more balanced set of DDIs to avoid skewing the model's predictions and using more interactions. Data from multiple databases could also be combined since DDI severity classifications can vary between sources. Embeddings with different numbers of characteristics could also be tested to see if longer or shorter vectors better describe the drug's network characteristics. Information about the chemical components and structure of each drug could be used in combination with these network characteristics so models can learn more about each drug. Other models could also be built to predict the mechanism of DDIs, the side effects, or the chance of an adverse event.

Acknowledgments

Thank you to the DDInter database for the data used in the project. A special thank you to Linda Banh for guidance and resources given throughout this project and the Inspirit AI team for the opportunity and resource support.

References

[1] Amidi, A., & Amidi, S. (n.d.). *Machine learning tips and tricks Cheatsheet Star*. CS 229 - Machine Learning Tips and Tricks Cheatsheet. Retrieved August 16, 2022, from https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

[2] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

[3] Cami, A., Manzi, S., Arnold, A., & Reis, B. Y. (2013). Pharmacointeraction network models predict unknown drug-drug interactions. *PLoS ONE*, *8*(4). https://doi.org/10.1371/journal.pone.0061468

[4] *Drug-Drug Interaction Network*. BioSNAP: Network datasets: Drug-drug interaction network. (n.d.). Retrieved August 16, 2022, from https://snap.stanford.edu/biodata/datasets/10001/10001-ChCh-Miner.html

[5] Ferdousi, R., Safdari, R., & Omidi, Y. (2017). Computational prediction of drug-drug interactions based on drugs functional similarities. *Journal of Biomedical Informatics*, *70*, 54–64. https://doi.org/10.1016/j.jbi.2017.04.021

[6] Foss, V. (2018). *Multiclass classification model evaluation*. Parasite ID. Retrieved August 16, 2022, from https://parasite.id/blog/2018-12-13-model-evaluation/#:~:text=We%20can%20understand%20a%20multiclass,to%20be%20a%20hookworm%20egg.

[7] Gandhi, R. (2018, July 5). *Support Vector Machine - introduction to machine learning algorithms*. Medium. Retrieved August 16, 2022, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[8] Grandini, M., Bagli, E., & Visani, G. (2021, February 14). *Metrics for multi-class classification: An overview*. – arXiv Vanity. Retrieved August 16, 2022, from https://www.arxiv-vanity.com/papers/2008.05756/

[9] Grover, A., & Leskovec, J. (2016). node2vec: Scalable Feature Learning for Networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/2939672.2939754

[10] Kumar, N. (2019, February). *Advantages and disadvantages of KNN algorithm in Machine Learning*. Advantages and Disadvantages of KNN Algorithm in Machine Learning. Retrieved September 2, 2022, from http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-knn.html

[11] Leung, K. (2021, September 1). *Network analysis and visualization of drug-drug interactions*. Medium. Retrieved August 16, 2022, from https://towardsdatascience.com/network-analysis-and-visualization-of-drug-drug-interactions-1e0b41d0d3df

[12] McKinney, W. AQR Capital Management, pandas: a python data analysis library, http://pandas.sourceforge.net

[13] McQuade, B. M., & Campbell, A. (2021). Drug Prescribing: Drug-Drug Interactions. *FP essentials*, *508*, 25–32.

[14] Open Classroom. (n.d.). *Exercise 5: Regularization*. Machine learning. Retrieved August 16, 2022, from http://openclassroom.stanford.edu/MainFolder/DocumentPage.php?course=MachineLearning&doc=exercises%2Fex5%2Fex5.html

[15] Palleria, C., Di Paolo, A., Giofrè, C., Caglioti, C., Leuzzi, G., Siniscalchi, A., De Sarro, G., & Gallelli, L. (2013). Pharmacokinetic drug-drug interaction and their implication in clinical management. *Journal of research in medical sciences : the official journal of Isfahan University of Medical Sciences*, *18*(7), 601–610.

[16] Rohani, N., & Eslahchi, C. (2019). Drug-drug interaction predicting by neural network using integrated similarity. *Scientific Reports*, *9*(1). https://doi.org/10.1038/s41598-019-50121-3

[17] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.

[18] Snap-Stanford. (n.d.). *Snap/examples/node2vec at master · snap-stanford/snap*. GitHub. Retrieved August 16, 2022, from https://github.com/snap-stanford/snap/tree/master/examples/node2vec

[19] Takeda, T., Hao, M., Cheng, T., Bryant, S. H., & Wang, Y. (2017). Predicting drug–drug interactions through drug structural similarities and interaction networks incorporating pharmacokinetics and pharmacodynamics knowledge. *Journal of Cheminformatics*, *9*(1). https://doi.org/10.1186/s13321-017-0200-8

[20] Vatsal. (2021, July 29). *Word2Vec explained*. Medium. Retrieved August 16, 2022, from https://towardsdatascience.com/word2vec-explained-49c52b4ccb71

[21] Vatsal. (2022, February 21). *Link prediction recommendation engines with node2vec*. Medium. Retrieved August 16, 2022, from https://towardsdatascience.com/link-prediction-recommendation-engines-with-node2vec-c97c429351a8

[22] Vatsal. (2022, January 31). *Node2vec explained*. Medium. Retrieved August 16, 2022, from https://towardsdatascience.com/node2vec-explained-db86a319e9ab

[23] Xiong, G., Yang, Z., Yi, J., Wang, N., Wang, L., Zhu, H., Wu, C., Lu, A., Chen, X., Liu, S., Hou, T., & Cao, D. (2021). DDInter: An online drug–drug interaction database towards improving clinical decision-making and patient safety. *Nucleic Acids Research*, *50*(D1). https://doi.org/10.1093/nar/gkab880

[24] Yan, C., Duan, G., Zhang, Y., Wu, F.-X., Pan, Y., & Wang, J. (2022). Predicting drug-drug interactions based on integrated similarity and semi-supervised learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *19*(1), 168–179. https://doi.org/10.1109/tcbb.2020.2988018