MSIS 5633 – Business Analytics Technologies

Section – In Class

Group Term Project

Predicting Severity of Vehicle Accidents with KNIME

Due Date:

May 5, 2024

By:

Rhoda Alawiye

Keevin Funderburg

Sai Suresh Marreddy

Oluwatoyin Oniroko

Table of Contents

MEET THE TEAM

Rhoda Alawiye

2nd Year MS BAnDS student with a background in Statistics

Keevin Funderburg

1st year MS BAnDS student at Oklahoma State University

In love with data!

Sai Suresh Marreddy
MS BAnDS 1st Year Student
Oklahome State University

- go-getter

Oluwatoyin Oniroko

1st year MS BAnD Student

Oklahoma State University

EXECUTIVE SUMMARY

In the United States, the alarming frequency of car crashes, at over 5.9 million police-reported crashes in 2022 underscores the critical need for continuous advancements in road safety. These accidents, ranging from minor injuries to fatalities, highlight the urgency of reducing both the frequency and severity of such incidents for the betterment of public health and safety. Since the pioneering introduction of seat belts in 1950 and the invention of airbags in 1951, the automotive industry has made significant strides in enhancing vehicle safety. Modern vehicles now boast an array of advanced safety features, including lane departure warnings and automatic braking systems, which represent monumental steps toward minimizing crash occurrences and severity.

Despite these advancements, the quest for zero accidents remains an ongoing challenge. It is within this context that our analysis plays a vital role. By utilizing detailed crash data provided by the National Highway Traffic Safety Administration (NHTSA), our study focused on meticulously analyzing over 23,000 car crash incidents to discern the critical factors that lead to severe injuries. Employing a suite of machine learning models, our team sought to isolate and identify the variables most predictive of high injury severity in crashes.

Vehicle travel speed emerged as the most critical factor, although imputation of half the speed values raises some uncertainty. Additionally, victim hospital transportation status was significant. With speed included, our best model, XGBoost, achieved 93% accuracy and 75% sensitivity; without speed, a random forest model achieved 83% accuracy with 43% sensitivity, and logistic regression achieved 76% accuracy with 75% sensitivity.

By turning complex big data into actionable intelligence, our project contributes to a data-driven approach to vehicle safety. We recommend the NHTSA tracks vehicle speed whenever possible, so such imputation does not have to be performed in the future. This report details the steps taken from initial data preparation to final model selection and discusses the implications of our findings in the broader context of road safety and preventive measures.

INTRODUCTION

The mission of the National Highway Traffic Safety Administration (NHTSA) is to "save lives, prevent injuries and reduce economic costs due to road traffic crashes, through education, research, safety standards and enforcement activity," (NHTSA Values). Every year in the United States, there are many car crashes that hurt people, damage property, and even take lives. According to the NHTSA, in 2022, there were an estimated 2.38 million people injured in motor vehicle traffic crashes (NHTSA, Overview). To help understand these crashes better, the NHTSA has created a tool called the Crash Report Sampling System (CRSS). CRSS collects information from police reports about all sorts of crashes, from small ones to big ones that cause serious injuries or deaths. This information helps NHTSA figure out where the most dangerous parts of the roads are and how to make them safer.

Can we use data from the CRSS to find what factors contribute to serious injuries resulting from vehicle-to-vehicle collisions? To answer this question, we will be conducting an analysis using multiple machine learning algorithms. This report follows the structure of the Cross-Industry Standard for Data Mining (CRISP-DM).

BUSINESS UNDERSTANDING

Over the years, the NHTSA has been improving the CRSS. From 2016 to 2021, the administration has expanded how it collects and codes data about crashes. CRSS works by sampling police-reported crashes of all types, ranging from crashes that only damage the vehicle to crashes involving fatalities (Crash Report 2023). Trained experts carefully read these reports and put the information into a computer system. Then, this data is checked to make sure it is correct and makes sense before it is released to the public (Crash Report 2023). CRSS shares this data with many groups, like government agencies, researchers, and the media, so everyone can work together to make driving safer for all (Crash Report 2023).

In 2022, the most recent year reports are available from, there were over 5.9 million police-reported crashes (NHTSA, Overview). Police-reported crashes with injuries and fatal crashes total over 1.7 million. (NHTSA, Overview). As mentioned in the introduction, the NHTSA

estimates a higher number of injuries than the police reported amount, at 2.38 million people (NHTSA, Overview).

With the NHTSA's mission in mind, part of preventing injuries can include learning what, if anything, leads to injuries in a crash. In the next section, we will explore the provided CRSS data and learn what exactly we can use to help predict how severe a crash's injury may be.

DATA UNDERSTANDING

The provided dataset consists of four distinct sets: Accidents, Vehicle, Person, and Distraction. These sets contain diverse attributes related to crash occurrences. To obtain a comprehensive overview of each accident, the four dataset sheets were merged using common columns 'case number' and 'vehicle number'. The segmented dataset utilized for our analysis was sourced from the CRSS analytical user's manual spanning from 2016 to 2021. This manual was obtained from a publication distributed by the NHTSA in 2023. The comprehensive dataset comprises of 23,354 rows and 34 columns, with the Injury severity (High / Low) serving as the target variable. Using these datasets, the team delved into a comprehensive analysis, extracting valuable insights into road accident patterns and underlying causes.

This study primarily aims at forecasting injury severity, with a particular interest in predicting the severity of injuries sustained by drivers. Despite the dataset encompassing various passenger categories, the team found the prediction of driver injury severity most intriguing. To streamline classification, injury severity was categorized into 'low' and 'high'. Numerous variables were provided, however, when we met as a team, we narrowed down these variables to what our prediction variable aim to target which was injury severity by understanding our factors and correlation with our target variable. **Figure 1** on the next page is part of the correlation matrix that aided our selection of variables utilized in prediction of the target variable:

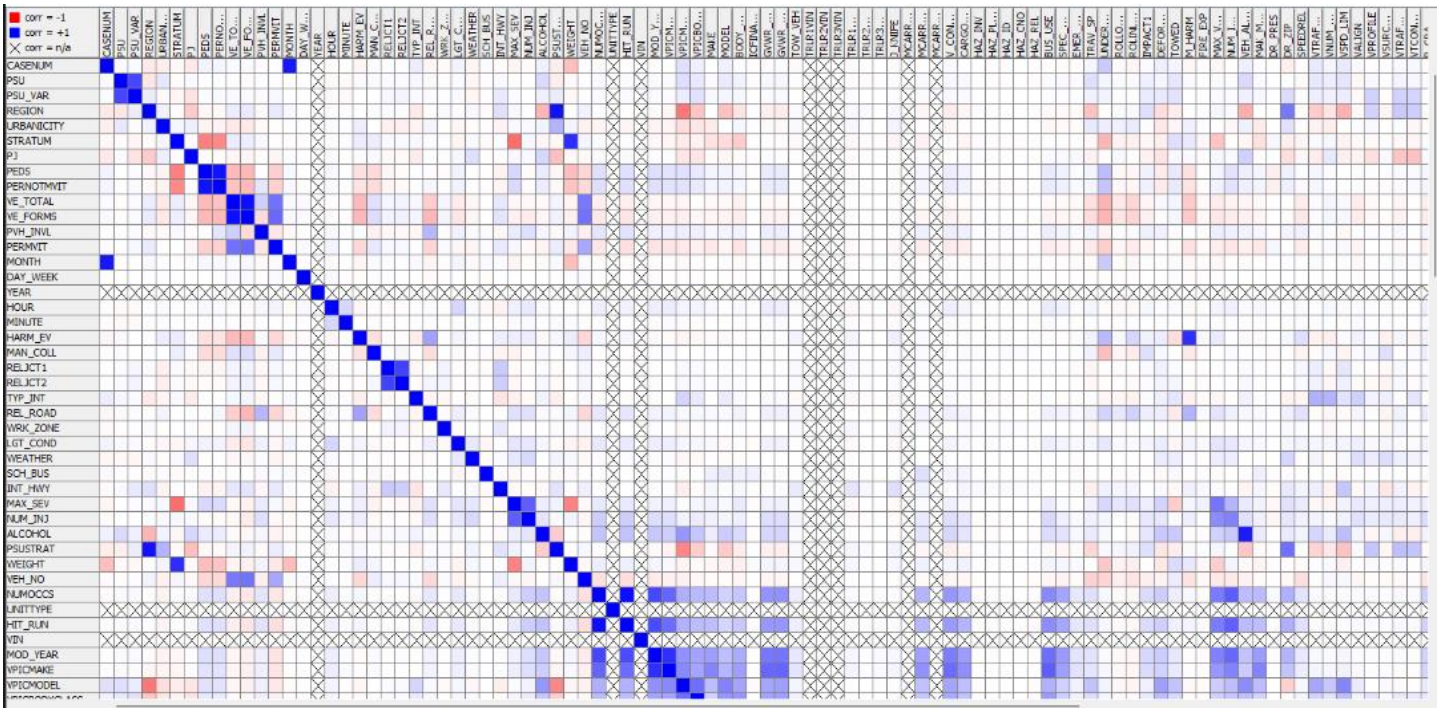

Figure 1: correlation matrix before data preprocessing

DATA PREPARATION

The data preparation phase is crucial in our analysis, ensuring the accuracy and efficiency of subsequent data analysis processes. This phase involves systematic procedures used to gather, clean, and organize four distinct datasets from the US National Highway Traffic Safety Administration (NHTSA). These datasets include real-world accident injury severity data which are critical for our analysis. Initially, the datasets: Accident, Vehicle, Person, and Distraction were merged using the Join nodes, utilizing their unique identifiers to ensure accurate alignment. To streamline the dataset for efficient analysis, duplicate columns were first removed. This step was critical in reducing the redundancy and enhancing the clarity of our data framework.

Subsequent filtering targeted the target variable “INJ_SEV” (Injury Severity) variable to focus on cases with high and low severity. This was accomplished using the numeric binner, setting specific bounds to isolate the targeted injury levels. Additionally, filters were applied to the “SEAT_POS” (Seat Position), “Body Type” and “Person Type” columns to refine the

dataset to relevant records, adhering to predefined conditions. Numerical variables were categorized into bins based on their respective criteria, as illustrated in **Figure 2** which entails details about the numerical binning for some of the selected variables.

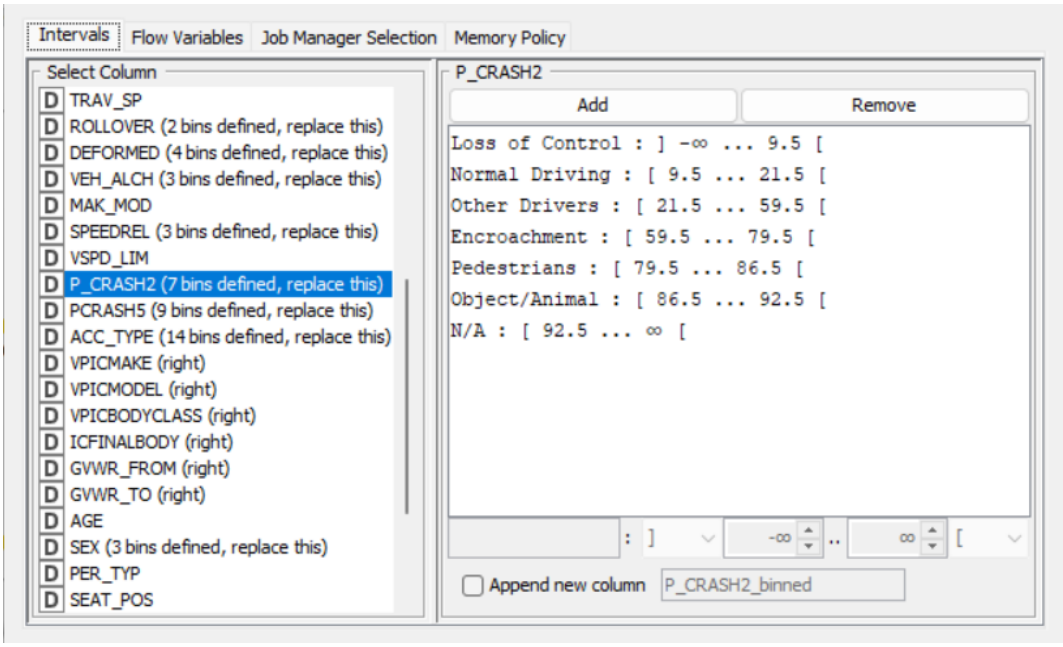

Figure 2: Numerical Binning Process

Multiple rule engine nodes were applied to non-binned variables such as “Age" column to restrict age groups to 120 years or less. Similar rules were applied to “Speed limit,” “Travel Speed” and “Model Year” columns to ensure data consistency and relevance. The target variable was split using the Row Splitter node to segment the data by Injury severity (“High”), which was then used for pattern matching and subset analysis. An independent group t-test was conducted to validate the split, showing significantly low p-value results for the “TRAV_SP” variable across the two injury severity categories, as shown in **Figure 3.** A concatenation node was used thereafter.

	Group	N	Missing Count	Missing Count (Group Column)	Mean	Standard Deviation	Standard Error Mean
TRAV_SP	Low	9777	9156	0	25.4644	23.0694	0.2333
TRAV_SP	High	1787	2634	0	42.7594	24.6004	0.5819

Levene Test
The Levene Test is used to test for the equality of variances.

	F	df 1	df 2	p-Value
TRAV_SP	2.4939	1	11562	0.1143

Independent Groups Statistics
Confidence Interval (CI) Probability: 95.0%
Differences are reported of the groups: Low - High

	Variance Assumption	t	df	p-value (2-tailed)	Mean Difference	Standard Error Difference	CI (Lower Bound)	CI (Upper Bound)
TRAV_SP	Equal variances assumed	-28.8365	11,562	1.22E-176	-17.295	0.5998	-18.4707	-16.1194
TRAV_SP	Equal variances not assumed	-27.5851	2,394.979	1.08E-145	-17.295	0.627	-18.5245	-16.0656

Figure 3: Independent Group t-test Result

Missing value in the "VSPD_LIM" (Vehicle Speed Limit) and Age were imputed using the median values, while a fixed year (2017) was assigned to the "MOD_YEAR" (Model Year) post concatenation. Mathematical nodes were employed to derive calculations for the "Speed over limit" and "Car age." Despite some variables showing moderate correlations, as depicted in the rank correlation matrix in **Figure 4,** they were retained due to their analytical importance. After preprocessing, the refined dataset comprised of 23,354 rows and 34 columns. The variables were categorized into numeric and nominal, with details provided in **Table 1** and **Table 2** respectfully.

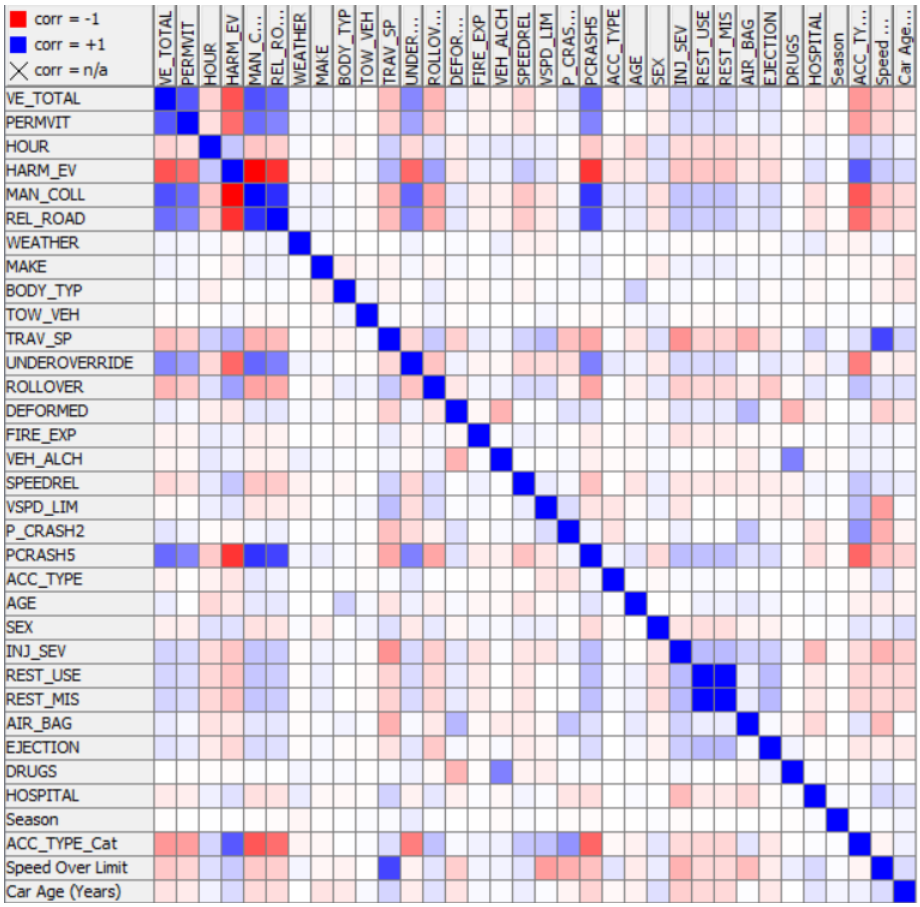

Figure 4: Rank Correlation Matrix for all selected variables

	Min	Max	Mean	St Dev	Number Zeroes
VE_TOTAL	1	13	2.073	0.789	0
PERMVIT	1	18	2.836	1.630	0
TRAV_SP	0	152	28.738	17.748	3,099
VSPD_LIM	0	80	44.055	12.876	367
AGE	12	99	40.822	17.519	0
Speed Over Limit	-75	107	-15.317	18.907	2,934
Car Age (Years)	-1	91	8.201	6.956	1,281

Table 1: Information about the numeric variables after preprocessing

Column	No. Missing	Unique Values	Column	No. Missing	Unique Values
HOUR	0	3	SPEEDREL	0	3
HARM_EV	0	4	P_CRASH2	0	7
MAN_COLL	0	3	PCRASH5	0	8
REL_ROAD	0	11	ACC_TYPE	0	11
WEATHER	0	3	SEX	0	3
MAKE	0	22	INJ_SEV	0	2
BODY_TYP	0	6	REST_USE	0	3
TOW_VEH	0	3	REST_MIS	0	3
UNDEROVERRIDE	0	4	AIR_BAG	0	3
ROLLOVER	0	2	EJECTION	0	3
DEFORMED	0	4	DRUGS	0	3
FIRE_EXP	0	2	HOSPITAL	0	3
VEH_ALCH	0	3	Season	0	4
			ACC_TYPE_cat	0	7

Table 2: Information about the nominal variables after preprocessing

Figure 7 on the next page illustrates the comprehensive data preprocessing workflow in KNIME. It is important to note that the N/A entries in nominal variables, marked during the binning phase, indicate “unknown” values rather than missing data.

This rigorous data preparation process has set a robust foundation for accurate and insightful analysis, ensuring that the data used reflects true, real-world conditions without bias or error.

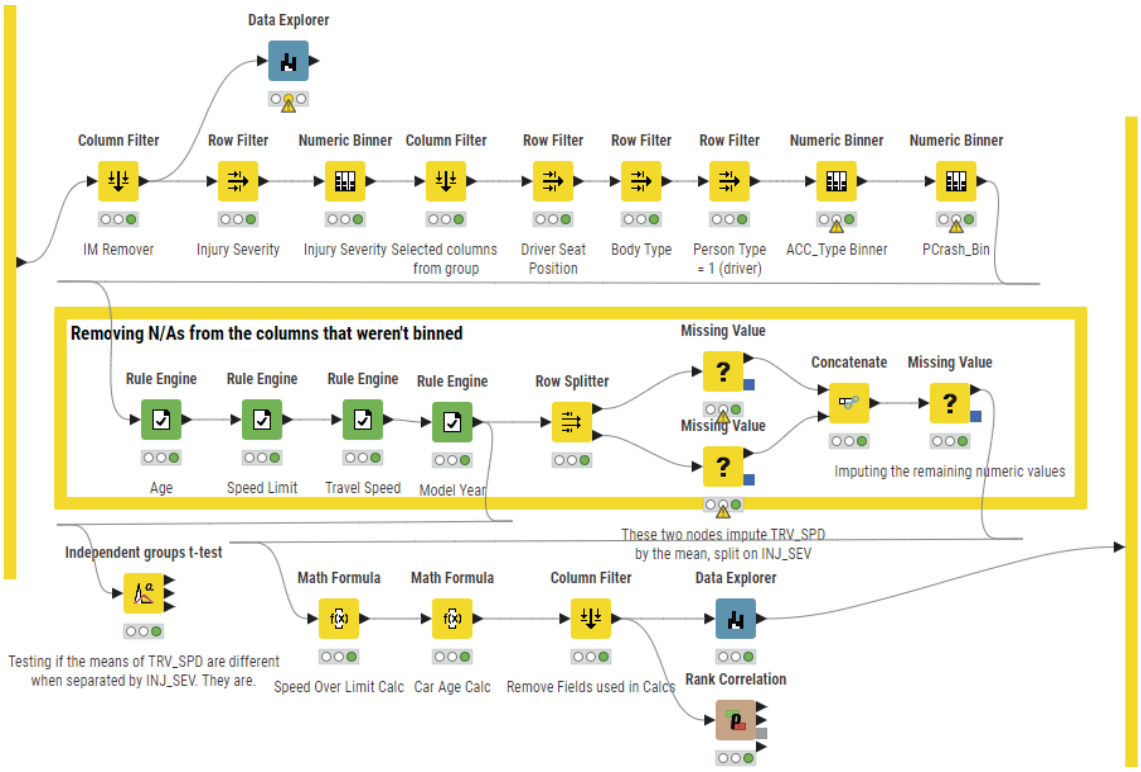

Figure 7: Data Preprocessing Workflow

MODELING

Based on the missing values and calculated fields, we built machine learning models in two ways. One includes Travel speed and speed over limit which is a calculated field and without those two variables. For all the models we used k-fold cross validation for training and testing purposes.

1. **Logistic Regression:** Logistic regression is a statistical method used for binary classification tasks. It predicts the probability of an event occurring based on one or more predictor variables. All the categorical columns are converted into numerical using one hot encoding which creates duplicates columns for each category. All the continuous variables are scaled using min max scalar. We used SMOTE for balancing data which oversamples the data using interpolation techniques.

i. **Without column filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	15,381	871	3,550	3,552	0.812	0.803	0.874	0.811
High	3,550	3,552	15,381	871	0.803	0.812	0.616	

Table 3a: Logistic Regression Statistics without Column Filter

ii. With Column Filter:

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	14,439	1,113	3,308	4,494	0.763	0.748	0.837	0.760
High	3,308	4,494	14,439	1,113	0.748	0.763	0.541	

Table 3b: Logistic Regression Statistics with Column Filter

This model is quite consistent with itself. The sensitivity and specificity are similar, so it is not a bad model at categorizing both low and high-level injuries.

2. **Decision Tree:** The main advantage of decision trees is how easy they are to interpret by humans. If one follows the rules of the tree from trunk to leaf, they can find exactly why an outcome is the way it is. Both of our decision trees split on the Gini index and

i. Without Column Filter:

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	17,345	810	3,611	1,588	0.916	0.817	0.935	0.897
High	3,611	1,588	17,345	810	0.817	0.916	0.751	

Table 4a: Decision Tree Statistics without Column Filter

As for the decision tree itself, it first splits on the travel speed. This correlates with what we found earlier, and with the variable importance we will show in the evaluation section in **Figure 12**.

Figure 8: Decision Tree without Column Filter

ii. With Column Filter:

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	15,094	1,620	2,801	3,839	0.797	0.633	0.847	0.766
High	2,801	3,839	15,094	1,620	0.633	0.797	0.506	

Table 4b: Decision Tree Statistics with Column Filter

As seen in **Figure 9** below, the first split is from the Hospital variable, which says whether a person was taken to the hospital or not. The next two variables to split are the number of people involved in the crash and whether a safety restraint was used or not.

Figure 9: Decision Tree with Column Filter

The decision tree is surprisingly accurate. It does have a noticeably lower sensitivity than specificity, and not only that, but the sensitivity went further down in comparison with the column filter applied.

3. **Naïve Bayes:** This type of model assigns probabilities of classification using Bayes' theorem. It assumes that the features are independent of one another.

   i. **Without Column Filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	15,335	1,574	2,847	3,598	0.810	0.644	0.856	0.779
High	2,847	3,598	15,335	1,574	0.644	0.810	0.524	

Table 5a: Naïve Bayes Statistics without Column Filter

ii. With Column Filter:

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	15,327	1,783	2,638	3,606	0.810	0.597	0.850	0.769
High	2,638	3,606	15,327	1,783	0.587	0.810	0.495	

Table 5b: Naïve Bayes Statistics with Column Filter

The sensitivity did not start particularly good and lowered to 0.587 with the column filter applied. Therefore, this model is not the best of the bunch at recognizing an elevated level of injury versus a low level.

4. **Random Forest:** This is another popular machine learning model used for classification and regression tasks. It also builds multiple decision trees and merges their outcome to improve decision accuracy and prevent overfitting. It handles a mix of numerical and categorical variables, providing insight into variable importance.

i. **Without column filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	18,389	1,111	3,310	544	0.971	0.749	0.957	0.929
High	3,310	544	18,389	1,111	0.749	0.971	0.800	

Table 6a: Random Forest Statistics without Column Filter

ii. **With Column Filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	17,591	2,526	1,895	1,342	0.929	0.429	0.901	0.834
High	1,895	1,342	17,591	2,526	0.429	0.929	0.495	

Table 6b: Random Forest Statistics with Column Filter

Notably, the sensitivity with the column filter went down quite a lot. 0.429 lowered to 0.749, which means that the model has a tough time correctly picking if an injury is severe.

5. **Gradient Boosted Trees:** Gradient Boosted Trees (GBT) is a popular machine learning technique known for its ability to decrease prediction errors iteratively. By sequentially building multiple decision trees, each one correcting the errors of its predecessor, GBT enhances model performance. This method is effective for both regression and classification tasks, as it captures complex data relationships. SMOTE is used to treat the imbalanced data which over samples the minor class using interpolation techniques.

   i. **Without Column Filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	18,193	983	3,438	740	0.961	0.778	0.955	0.926
High	3,438	740	18,193	983	0.778	0.961	0.8	

Table 7a: GBT Statistics without Column Filter

   ii. **With Column Filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	16,941	2,047	2,374	1,992	0.895	0.537	0.893	0.827
High	2,374	1,992	16,941	2,047	0.537	0.895	0.54	

Table 7b: GBT Statistics with Column Filter

The GBT overall has the highest accuracy we have seen so far, at 92.6%. There is again the same issue with sensitivity lowering with the column filter.

6. **XGBoost Tree Ensemble:** "XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable," (XGBoost, 2022). In testing, it tends to have satisfactory results very comparable to the GBT.

   i. **Without Column Filter:**

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	18,391	1,098	3,323	542	0.971	0.752	0.957	0.930
High	3,323	542	18,391	1,098	0.752	0.971	0.802	

Table 8a: XGBoost Statistics without Column Filter

ii. With Column filter:

	True Positive	False Positive	True Negative	False Negative	Sensitivity	Specificity	F-measure	Accuracy
Low	17,534	2,524	1,897	1,399	0.926	0.429	0.899	0.832
High	1,897	1,399	17,534	2,524	0.429	0.926	0.492	

Table 8b: XGBoost Statistics with Column Filter

Finally, the XGBoost Tree Ensemble had the highest accuracy of all, with 93%. However, the sensitivity of both versions of the model are lower than that of the GBT.

EVALUATION

To compare all the models with each other, we have created **Table 9a** and **Table 9b**:

	Confusion Matrices			Accuracy	Sensitivity	Specificity
		Low	High			
Logistic Regression	Low	15,381	3,552	0.811	0.803	0.812
	High	871	3,550			
Decision Tree	Low	17,345	1,588	0.897	**0.817**	0.916
	High	810	3,611			
Naïve Bayes	Low	15,335	3,598	0.779	0.644	0.810
	High	1,574	2,847			
Random Forest	Low	18,389	544	0.929	0.749	**0.971**
	High	1,111	3,310			
Gradient Boosted Trees	Low	18,193	740	0.926	0.778	0.961
	High	983	3,438			
XGBoost	Low	18,391	542	**0.930**	0.752	**0.971**
	High	1,098	3,323			

Table 9a: Model Comparison without Column Filter

Without the column filter, the XGBoost tree ensemble had the best accuracy and specificity, with the random forest at a very close second. However, the regular decision tree had the best sensitivity, with the logistic regression not so far away.

	Confusion Matrices			Accuracy	Sensitivity	Specificity
		Low	High			
Logistic Regression	Low	14,439	4,494	0.760	**0.748**	0.763
	High	1,113	3,308			
Decision Tree	Low	15,094	3,839	0.766	0.633	0.797
	High	1,620	2,801			
Naïve Bayes	Low	15,327	3,606	0.769	0.587	0.810
	High	1,783	2,638			
Random Forest	Low	17,591	1,342	**0.834**	0.429	**0.929**
	High	2,526	1,895			
Gradient Boosted Trees	Low	16,941	1,992	0.827	0.537	0.895
	High	2,047	2,374			
XGBoost	Low	17,534	1,399	0.832	0.429	0.926
	High	2,524	1,897			

Table 9b: Model Comparison with Column Filter

Interestingly, the regular random forest had a better result than the more recent boosted algorithms with the column filter. And the simplest algorithm, logistic regression, had the best sensitivity with the column filter. However, the logistic regression had the worst accuracy and specificity out of them all with the column filter.

Taking these winning models both with and without the column filter, we put them into a ROC curve to compare them. This curve can be seen on the next page in **Figure 10**. While it is unsurprising that the models without the column filter run better than the others, it is interesting how close the regular decision tree is to the boosted and bagged trees. The area under the curve of the XGBoost is the highest at 0.969. With the filter, the best is the logistic regression with a 0.846 area under the curve.

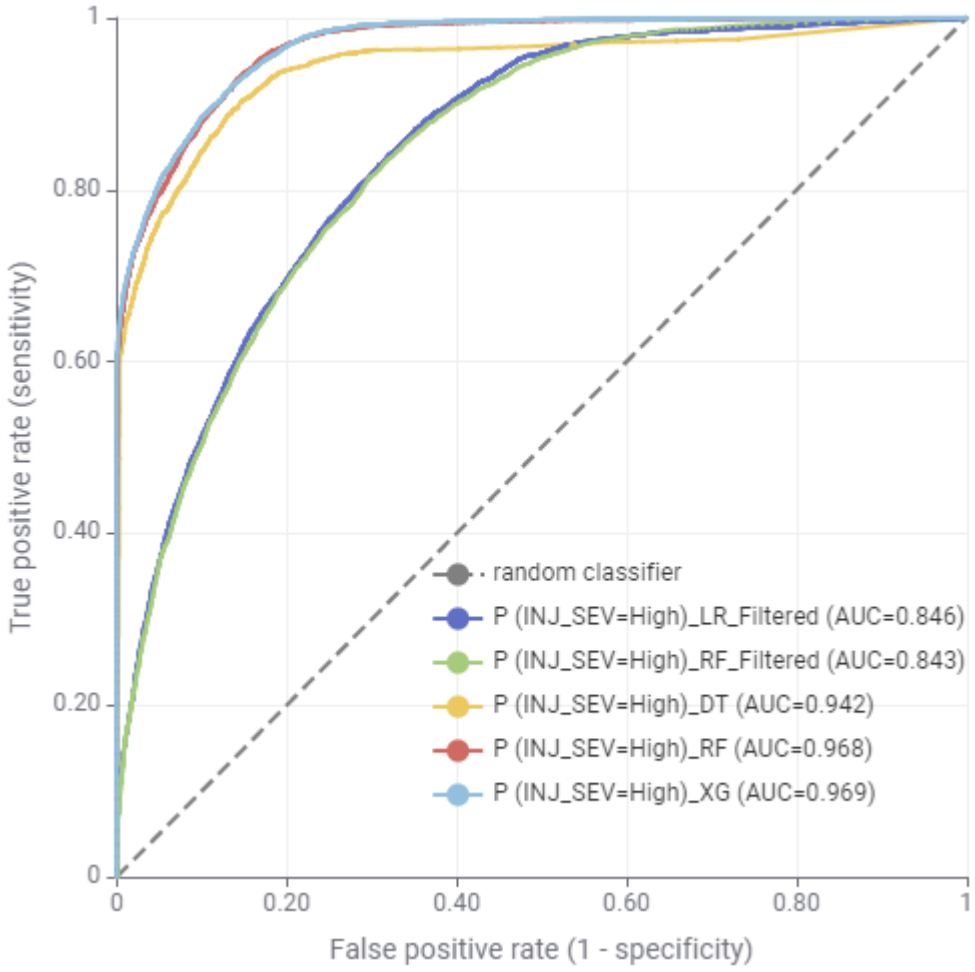

Figure 10: ROC Curve Comparison

The random forest can output a variable importance, as can XGBoost. For the first, we used a math formula node to multiply the splits by the candidates, with rank 1 having a weight of 0.4 and rank 2 having a weight of 0.1.

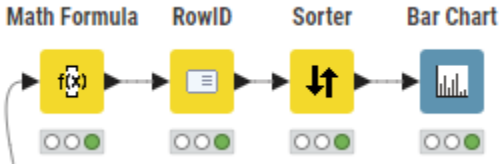

Figure 11: Nodes to Make a Random Forest Variable Importance Chart

Both random forest models were run, meaning we have an output with the filter and without. **Figure 12** shows the variable importance from the unchanged dataset compared to the changed dataset.

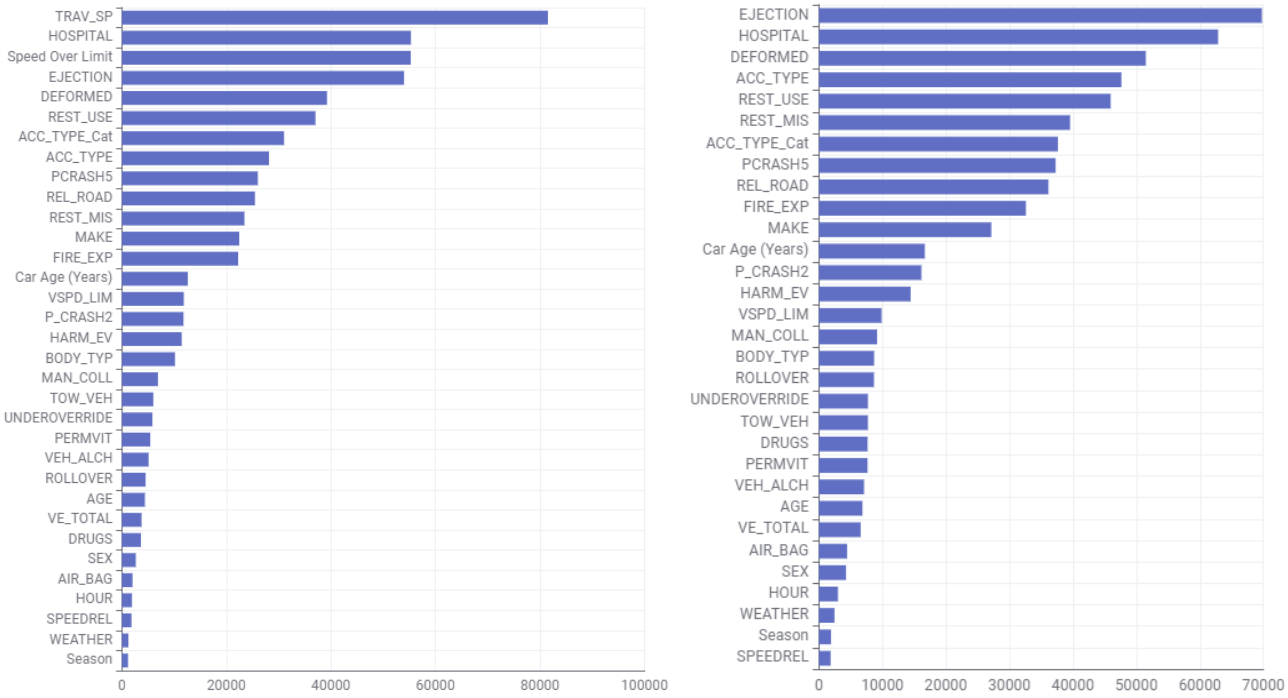

Figure 12: Variable Importance without and with Column Filter

There are a few interesting differences between the two. When the travel speed and the speed over limit are removed, Ejection becomes more important than if the victim was taken to the hospital or not. Restraint misuse also jumps up, from 11th place to 6th. But for the most part, the variables which are most important are around the same place in both charts.

XGBoost outputs weight, gain, and cover for its feature importance. For the purpose of this analysis, we only compared the gain of the model. Unlike the random forest, it showed which categories in the categorical variables were most important. It is all shown in **Figure 13** on the next page.

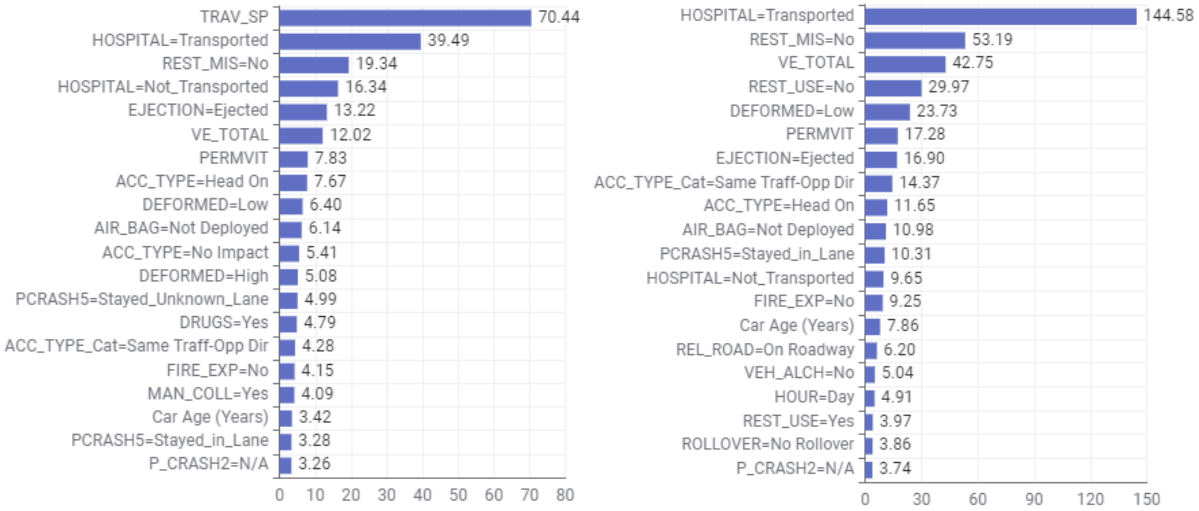

Figure 13: Top 20 XGBoost Variable Importance without and with Column Filter

What model should we choose; which approach is best? The answer to these questions depends on the use case. We explore this further in the conclusion.

DEPLOYMENT

In this academic project, we focused on the theoretical aspects of deploying a model, which is a crucial step in transitioning from a development environment to practical, real-world applications. Although direct implementation in live environments was beyond our scope, understanding this process is essential for ensuring the readiness of our models for actual deployment.

Our approach to deployment emphasized the importance of version control using systems like GitHub to manage and document our model's development. This ensures transparency and the ability to revert to earlier versions if necessary.

While KNIME Analytics Platform is open source, the company also offers a paid solution, KNIME Business Hub. The Business Hub offers version control, much like a service such as GitHub (KNIME, Business). One of KNIME Business Hub's use cases is continuous deployment: they offer "a standardized (but fully customizable) validation and release test process," (KNIME, Continuous).

If we were to deploy these models for a company which needs them to be continuously monitored, the KNIME Business Hub would be a good solution. However, its pricing begins at $39,900 per year, which may not be worthwhile if the company does not use KNIME for other projects (KNIME Pricing).

We also investigated the concept of containerization, such as using Docker, which packages the model with all its dependencies, ensuring consistent performance across different computing environments (Docker).

Furthermore, we explored the theoretical use of orchestration tools like Kubernetes, which would automate the deployment and management of containerized applications, ensuring efficient resource use and scalability. Monitoring is highlighted as a crucial ongoing process post-deployment, to ensure the model performs well and stays relevant as conditions change.

By integrating these concepts into our project, we aim to prepare for real-world data science applications, ensuring our models are robust, scalable, and maintainable when deployed in actual settings.

CONCLUSION

Our exploration into the predictive modeling of injury severity from vehicle crashes has been a comprehensive journey, starting from data preparation to the final model evaluation. Throughout this process, our team focused on identifying and leveraging the most impactful variables using advanced machine learning techniques. The XGBoost model proved the most robust if all variables were kept. However, if the vehicle travel speed were removed, the best model was either the logistic regression, with a good accuracy and sensitivity, or a random forest with higher accuracy and low sensitivity.

In future data collection, we recommend the NHTSA makes sure that crash reports in the CRSS include the vehicle travel speed whenever possible. Imputation such as what we performed should not have to be done, especially when the variable ended up becoming the most important.

This project highlighted the importance of meticulous data preprocessing, insightful model selection, and rigorous evaluation in building effective predictive models. While our academic setting provided a controlled environment for this exploration, the practices and insights gained are universally applicable, preparing us for real-world data science challenges.

As we look forward, the dynamic nature of technology and data science promises new tools and methodologies that will enhance our ability to model and predict outcomes even more accurately. However, the foundational skills and knowledge we have developed during this project will remain indispensable, equipping us to contribute meaningfully to the ongoing efforts in enhancing road safety through data-driven insights and innovations.

REFERENCES

Crash Report Sampling System Analytical User's Manual, 2016-2021 (2023). National Highway Traffic Safety Administration.

Delen, D., Tomak, L., Topuz, K., & Eryarsoy, E. (2017). Investigating injury severity risk factors in automobile crashes with predictive analytics and sensitivity analysis methods. *Journal of Transport & Health, 4*, 118-131. https://doi.org/10.1016/j.jth.2017.01.009

Docker (2024). Docker Inc. https://www.docker.com

KNIME Business Hub (2024). KNIME. https://www.knime.com/knime-business-hub

KNIME for Continuous Deployment of Data Science (2024). KNIME. https://www.knime.com/solutions/knime-continuous-deployment-data-science

KNIME Hub Pricing (2024). KNIME. https://www.knime.com/knime-hub-pricing

NHTSA's Core Values (n.d.). National Highway Traffic Safety Administration. https://www.nhtsa.gov/about-nhtsa/nhtsas-core-values

Overview of Motor Vehicle Traffic Crashes in 2022 (2024). National Highway Traffic Safety Administration. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813560

XGBoost Documentation (2022). dmlc XGBoost. https://xgboost.readthedocs.io/en/latest/index.html