ANALYSIS OF THE EFFECT OF COVID ON THE PAYMENT OF HEALTHCARE EXPENSES AND PAYMENT PREDICTION USING MACHINE LEARNING ALGORITHMS

Author:

Sravani Mahankali

Department:

Department Of Health Data Science, Saint Louis University - School of Medicine

Course:

HDS 5960 Health Data Science Capstone

Supervisor:

Sylvia Neil, Senior Director Business Intelligence

Instructors:

Divya S. Subramaniam, Ph.D., Mph
Assistant Professor and Program Director
Department Of Health and Clinical Outcomes Research
School Of Medicine

Paula Buchanan, Ph.D., Mph
Associate Director for Consulting, Advanced Health Data (Ahead) Institute
Associate Professor and Associate Director of Academic Affairs
Department Of Health and Clinical Outcomes Research
School Of Medicine

Srikanth Mudigonda, Ph.D.
Associate Professor and Interim Associate Dean for Academic Affairs
Director Of Applied Analytics
School For Professional Studies
Saint Louis University

Deepika Gopukumar, Ph.D.
Assistant Professor
Department Of Health and Clinical Outcomes Research
School Of Medicine

Date:
05/05/2023

ABSTRACT

This study mainly analyses the effect of covid on healthcare payments and predicts if the claims are paid or not. This study utilized the Dask framework in Python and claims and denial data from Envision Healthcare to perform statistical and predictive analyses. The datasets were cleaned by removing nulls and changing the data type of attributes to the appropriate data types. Descriptive statistics such as count, mean, standard deviation, minimum value, maximum value, and interquartile range were calculated for numerical attributes, and categorical attributes were changed to the categorical data type. Visualizations including scatter plots, heatmaps, stacked bar charts, and bar charts were used to identify interactions and correlations between variables. Statistical tests such as the Chi-Squared test and Multinomial regression were used to analyze trends in pre- and post-COVID data, and to analyze the relationships between types of insurance, claim status, and time taken for payment. Supervised machine learning classification models such as Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost, and XG Boost were used to predict whether claims were paid, and a linear regression model was fit to predict the time taken for payment. The statistical analysis showed that there is a relationship between relationship between the type of insurance and claim status category, and that there was a relationship between the type of insurance and the number of days taken for payment. The study also found a relationship between claim status category and billing provider state, with a percentage change in the number of days from pre-COVID to post-COVID. In the predictive analysis it is found that the best model for predicting payment of claims was Random Forest.

Key Words - Predictive Analysis, Statistical Analysis, Pre-Covid, Post-covid, Supervised Machine Learning

I. INTRODUCTION

I have completed my capstone under Sylvia Neil, Director of Business Intelligence & Analytics at Envision HealthCare based in Fort Lauderdale, Florida. Envision Healthcare is a national medical group having two main services, one is the multispecialty physician group and healthcare management team called Envision Physician Services and AMSURG which acquires, develops, and operates ambulatory surgery centers (ASC) in partnership with physicians throughout the United States. This project is specifically useful for the organization to analyze various aspects related payments for the years 2018-2022, predict if the payments will be made by the patients and manage the healthcare costs overall.

The COVID-19 pandemic has had a significant impact on healthcare systems worldwide, with many countries experiencing a surge in demand for healthcare services and an increase in healthcare expenses [1]. As the pandemic majorly continued till 2022, healthcare providers and policymakers are still faced with the challenge of managing healthcare costs while ensuring access to high-quality care for all patients. In this context, the use of machine learning algorithms for payment prediction has gained increasing attention. Machine learning algorithms offer the potential to improve the accuracy of payment prediction, thereby improving cost-effectiveness and managing healthcare costs [2]. This article aims to analyze the effect of COVID-19 on the payment of healthcare expenses and explore the potential of machine learning algorithms for payment prediction in the context of the pandemic. The article will review existing literature on the impact of COVID-19 on healthcare expenses and discuss the potential of machine learning algorithms for predicting payments. The findings of this study could have important implications for healthcare providers and policymakers, as they seek to manage healthcare costs and payments in the context of the COVID-19 pandemic. The study could also contribute to a better understanding of the potential of machine learning algorithms for payment prediction.

A study closely related to this project presents a set of rules for predicting medical claims payments based on various factors such as the type of service provided, location, and provider's credentials and the authors developed a prototype system based on these rules and evaluated its performance on a dataset of Medicaid claims (Wojtusiak, Ngufor, Shiver, Ewald, 2011). The results showed that the rule-based approach which internally used machine learning models achieved high accuracy in predicting Medicaid claims payments compared to traditional machine learning models and the method can detect irregularities in payments. The article also provides a valuable perspective on the potential of a rule-based approach for predicting medical claims payments and improving revenue cycle management in healthcare organizations, with an initial application to Medicaid data (Wojtusiak, Ngufor, Shiver, Ewald, 2011). Though not very similar another study uses data mining techniques to predict to identify claims with a high probability of errors before they were processed and prevent errors in health insurance claims processing (Kumar, Ghani, Mei Z, 2010). In another related work the authors have described a rule-based approach called "predictive analytics" to classify hospital payments and mentioned the use of automation of the analysis of patient accounts to ensure charges are submitted (Bradley, Kaplan, 2010). The authors have also highlighted the use of predictive analysis to help detect under payments (Bradley, Kaplan, 2010). Overall, the article highlights the potential benefits of using data analytics to improve financial performance in hospitals and suggests that data-driven decision-making is essential for successful healthcare management (Bradley, Kaplan, 2010). Furthermore, another article highlights the importance of statistical methods in healthcare fraud detection, noting that manual review of healthcare claims is time-consuming and may miss fraudulent claims (Li, Huang, Jin, Shi, 2008). The authors suggest that statistical methods can be used to complement manual review and increase the efficiency and accuracy of fraud detection, discusses the challenges and limitations of using statistical methods for healthcare fraud detection, including the potential for false positives and the need for ongoing monitoring and refinement of detection methods (Li, Huang, Jin, Shi, 2008). The author of another article proposes two models for detecting fraudulent claims within unsupervised databases and notes that the models can identify both known and unknown fraudulent claims and can be used to identify patterns and trends in fraudulent behavior

(Musal, 2010). The article highlights the potential of unsupervised machine learning algorithms in detecting healthcare fraud, noting that these methods can analyze large datasets and identify patterns that may be difficult to detect using manual review (Musal, 2010).

An article “Turning Hospital Data into Dollars” by Bradley, Kaplan, 2010 used a rule-based approach called “predictive analytics” to classify hospital payment, in the same way this study will be using predictive analysis however, this goes beyond just classification, this study predicts if the payments will be paid or not. Although the article "Rule-Based Prediction of Medical Claims' Payments: A Method and Initial Application to Medicaid Data" by Wojtusiak, Ngufor, Shiver, Ewald, 2011 used a rule-based approach to make predictions on Medicaid claims payments, it does not provide an intuition on how the rule-based approach works for all the types of insurance claims and insurance denials. This gap will be filled by this study where the data related to all types of insurance claims and denials will be used for statistical analysis in the context of Covid19 and building various supervised machine learning models. Another article “Two models to investigate Medicare fraud within unsupervised databases. Expert Systems with Applications” by Musal, 2010 has suggested the use of unsupervised machine learning algorithms in detecting the fraud claims in medicare data, and article “A survey on statistical methods for health care fraud detection. Health Care Management Science” by Li, Huang, Jin, Shi, 2008 has also suggested the use of statistical methods for fraud detection for increased accuracy. Though this article does not detect the fraud payments directly, this uses supervised machine learning algorithms for payment prediction on all kinds of data.

This study analyses the types of insurances accepting or denying the insurance claim for the payment of healthcare expenses, the time taken for the payment, state wise analysis of the payments Pre-Covid and Post-Covid era. Which in turn helps to discover the effect of Covid on the payment of healthcare expenditures, and finally predict whether the unknown and future records would be paid or not. This study uses two datasets, firstly they are cleaned then goes on to calculation of descriptive statistics, data visualization, performing statistical tests such as t-test, ANOVA, Chi-SQ test, correlation test, including the alternate tests for each one of the tests. Finally, it uses a set of supervised machine learning models such as Logistic regression, linear regression, Gradient descent boost method, Random forests, Support vector machines, K-nearest neighbors. This study also evaluates the predictive power of supervised machine learning algorithms based on accuracy.

II. MATERIALS AND METHODS

Software and Dataset Used

The software used in this project is the Dask framework in Python, and the data set used is the claims and denial data from the database of the company I work in (Envision Healthcare). The code is written in a jupyter notebook.

Data Cleaning

The nulls are removed from the dataset, the data type of the attributes are changed to the appropriate data types. The attributes (Statement End Date, Statement Start Date, Claim Received Date) related to dates are changed to datetime, and the attributes (Claim Status Category, RMA_Payer: Prim Lvl III, Denial Type, Claim Status Description, Claim Status Category, Billing Provider State, Billing Provider City) related to categories are changed to categorical data type in remits dataset. Similarly, the attributes (Statement End Date, Statement Start Date, Claim Bill Date) related to datetime and the attributes (Billing Provider City, RMA_Payer: Prim Lvl III, Billing Provider State) related to categorical are changed to their respective data types in claims dataset. The attribute Admit Type is removed from the dataset as it is entirely null.

Descriptive Statistics

The Descriptive statistics such as count mean, standard deviation, minimum value, maximum value, inter quartiles are calculated for the numerical attributes in both the remits dataset (Statement End FY, Initial Denial Remit Count, Total Denied Remit Count, Final Denials Remit Count) and claims dataset (Statement End FY, Claim Count).

Data Visualization

The visualizations namely scatter plots, heatmaps, stacked bar charts, and bar charts on both the datasets i.e., Claims and remits datasets. The data visualization charts has revealed interactions between RMA_Payer: Prim Lvl III and Claim Status Category, as well as RMA_Payer: Prim Lvl III and Billing Provider State. Scatter plots revealed the extent linearity between the numerical variables, and heatmaps have revealed the extent of correlation.

Statistical Analysis

To analyze the trends in pre-covid and post-covid era, the remits dataset is divided into two parts based on the date. The records before 01-01-2020 are included in the pre covid dataset and the records after 01-01-2020. The Statistical tests used in this study on the divided datasets are mainly Chi-Squared test and Multinomial regression, the tests were chosen based on the type of variable (categorical or quantitative).

Analysis of types of insurances accepting or denying the claims:

Firstly, chi-squared test is performed to know whether there is a relationship between the type of insurance and the claim status category The outcome variable is RMA_Payer: Prim Lvl III) and the predictor variable is claim status category, both are categorical variables. Also, the assumptions of chi-squared test namely, all observations should be independent, all the variables should be nominal or ordinal, and the expected values should be 5 or higher in at least 80% of groups are checked. Then the percentages of denied and accepted claims for each type of insurance.

Analysis of time taken for the payment based on the payer type:

The Multiple Linear Regression is performed to analyze whether there is a relationship between the type of insurance and the number of days. The outcome variable is RMA_Payer: Prim

Lvl III and the predictor variable is number of days. Also, the assumptions of multinomial regression namely independence of observations, linearity, no perfect multicollinearity are checked.

Analyzing the claim status and number of days needed for claim reimbursement based on state:

To perform state wise analysis, the chi-squared test is performed to analyze the relationship between Claim Status Category and Billing Provider State. Also, the assumptions of chi-squared test namely, all observations should be independent, all the variables should be nominal or ordinal, and the expected values should be 5 or higher in at least 80% of groups are checked. Then the effect of state on the claim status and percentage change in number of days from pre-COVID to post-COVID are visualized.

Predictive Analysis

Predicting if the claims are paid or not:

Various supervised machine learning classification models namely, Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost, XG Boost are used to predict if the claims are paid or not as the outcome variable is categorical. Accuracy and Time is also computed for each of the models to identify the best model for predicting the payment of claims, the data set used is the remits data set. The outcome variable in the predictive analysis is Claim Status Category and the predictor variables are Statement End FY, RMA_Payer: Prim Lvl III, Claim Status Description, Denial Type, Claim Filling Code, Billing Provider State, Billing Provider City, Initial Denial Remit Count, Total Denied Remit Count, Final Denials Remit Count, num_days. The outcome variable is converted to binary variable encoding the denied category as 1 and rest as 0. The predictor variables are also label encoded before fitting the models.

Predicting time taken to make the entire payment:

A linear regression model is fit to predict the time taken for the payment as the outcome variable (num_days) here is numerical. The model is first fit on the remits data set and the score is calculated then the same model is applied on the claims data set considering it as the new records. The predictor variables are RMA_Payer: Prim Lvl III, Billing Provider State, Billing Provider City and the outcome variable is num_days.

III. RESULTS

Descriptive Statistics

The Descriptive statistics are calculated on both the datasets (claims dataset, remits dataset). There are 516964 total records in the remits dataset, where initially 34 remits are denied at maximum and 28 the final denial remit count. However, most of the claim counts are received after the third quartile (75%). The maximum total remits count is 55. The maximum number of days needed to complete the payment of claims is 8172. The results are tabulate in Table 1. The maximum number of claims received from insurance companies is 52 claims in the claims data set.

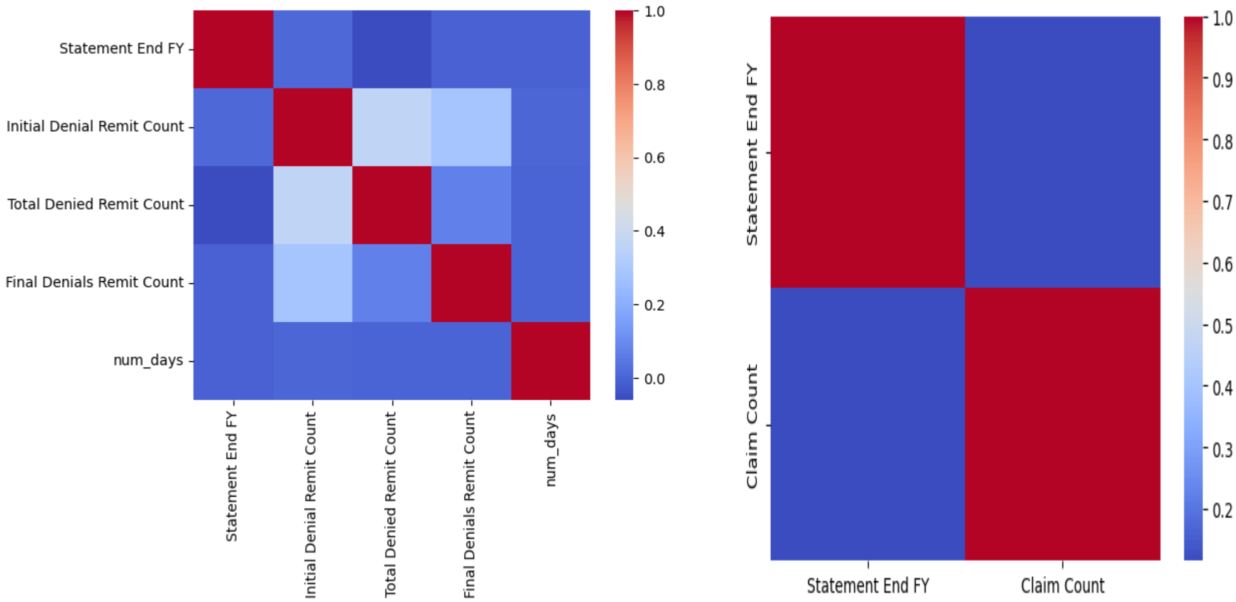
Table 1: Descriptive statistics of Remits Data

Statistics	**Statement End FY**	**Initial Denial Remit Count**	**Total Denied Remit Count**	**Final Denials Remit Count**	**num_days**
count	516959	516959	516959	516959	516959
mean	2020.657	0.196577	0.676858	0.016688	0.314354
std	0.968968	0.434827	0.967532	0.136818	21.397328
min	2018	0	0	0	0
25%	2020	0	0	0	0
50%	2021	0	1.0	0	0
75%	2021	0	1.0	0	0
max	2022	34.0	55.0	28.0	8172.0

Data Visualization

While visualizing the remits dataset, it is seen that most of the records in the data set are Medicaid payers and very few are from self-pay. From the scatter plots it is found that there is no linearity between the number of days needed for the payment and the denial remit count, and there is a correlation between the denial remit count (Fig 2(b)). From the correlation plots here is a slight correlation between year and initial denial remit count (Fig 1(a)). From the stacked bar plot, it is observed that most of the Medicare payments are processed i.e., the claims are retrieved. Among all the payers, Medicaid insurances are the most denied ones. Among all the payers the commercial payers had more claims which are reversed. None or very few claims are forwarded to other payers among all types of payers (Fig 3(a)). It is also found that most of the processed claims are in the year 2021 and the least are in 2018. Most of the claims are denied in the year 2021, and very few claims are denied in 2018 (Fig 3(b)).

In the claims dataset, medicare had more records and less self-pay. From scatter plots it is observed that there is no linearity between the claim count and year(Fig 2(a)). From the stacked bar charts, among all other states, Florida has the most records in all the payer types and among those Medicare has the most records. Other states having the most records in all payers is Texas and Tennessee (Fig 4(a)). It is also observed that most of the payers had a count below 10 claims (Fig 4(b)). Moreover, from heatmaps it is clear that there is no correlation between the claim count and year (Fig 1(b)).



(a) **(b)**

Fig 1: heatmaps (a) Heatmap of Remits Dataset (b) Heatmap of Claims Dataset

Table 2: (a) Proportional Change of accepted and denied claims for each provider in pre-COVID era

RMA_Payer: Prim Lvl III	Claim Status Category	Proportion Change
Commercial	Denied	6.274348
	Forwarded to Other Payer	0.105413
	Processed	85.384298
	Reversal	8.235941
Medicaid	Denied	33.509458
	Forwarded to Other Payer	0.021135
	Processed	59.019338
	Reversal	7.450069
Medicare	Denied	4.847611
	Forwarded to Other Payer	0.003066
	Processed	91.773472
	Reversal	3.375851
Self-Pay	Denied	20.595383
	Forwarded to Other Payer	0.000000
	Processed	69.623329
	Reversal	9.781288

Table 2: (b) Proportional Change of accepted and denied claims for each provider in post-COVID era

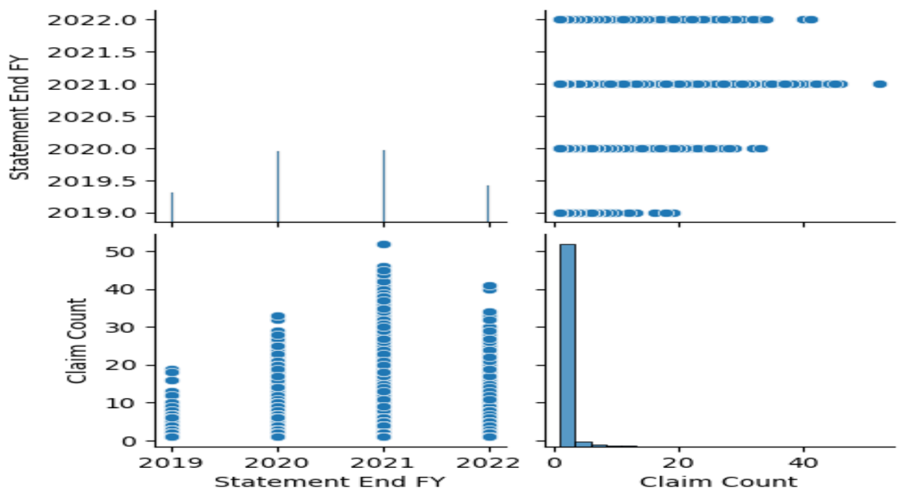
RMA_Payer: Prim Lvl III	Claim Status Category	Proportion Change
Commercial	Denied	8.297133
	Forwarded to Other Payer	0.135828
	Processed	84.963473
	Reversal	6.603566
Medicaid	Denied	21.070087
	Forwarded to Other Payer	0.189302
	Processed	74.094378
	Reversal	4.646232
Medicare	Denied	4.249054
	Forwarded to Other Payer	0.005811
	Processed	92.889832
	Reversal	2.855303
Self-Pay	Denied	10.265372
	Forwarded to Other Payer	0.019417
	Processed	85.682848
	Reversal	4.032362

Statistical Analysis

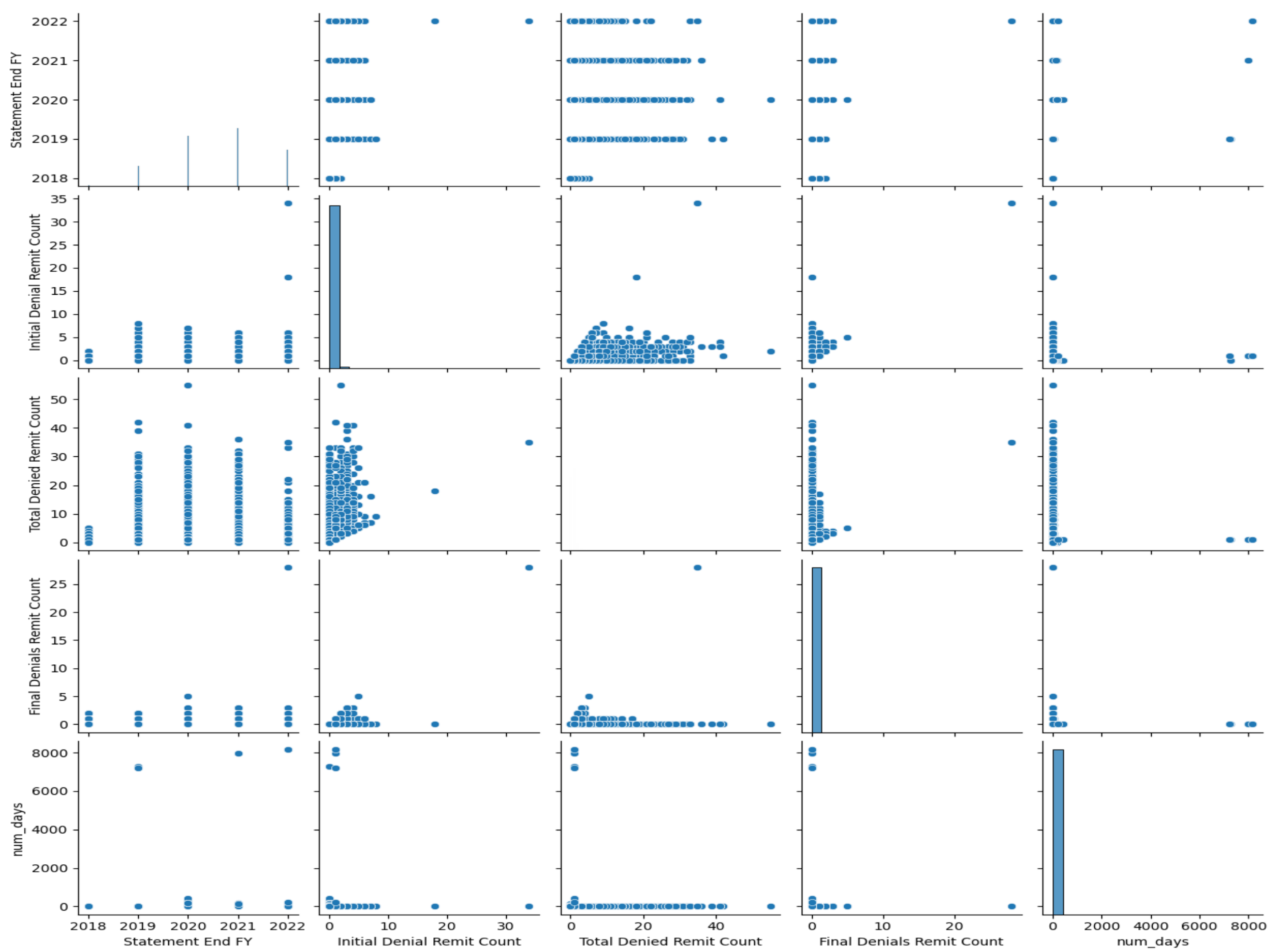
Analysis of types of insurances accepting or denying the claims:

From the chi-square test it is revealed that there is a relationship between Claim Status Category and the RMA_Payer: Prim Lvl III as the p-value is less than 0.05. When the standardized residuals is performed to check which cell is contributing towards the chi-squared statistic, it is observed that all the standardized residual values are between -2 and 2 indicating that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic. The odds ratios equal to 1 reveal that there is slightly a positive association between all the groups of claims status category and RMA_Payer: Prim Lvl III.

All the assumptions of the chi-squared test are met. Specifically, since all the observations are collected independently, the assumption all observations are independent is met. As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met. The assumption of the expected values should be 5 or higher in at least 80% of groups is met. These results are the same throughout the years i.e., there is no fluctuations in the pre-covid and post-covid Era. However, from the calculations of percentages it is seen that the denial of commercial claims increased by 2% but the self-pay denial decreased by 10%. Interestingly the payment trend was positive during the post-covid period. The result of the proportional change is tabulated in Table 2(a) and 2(b).



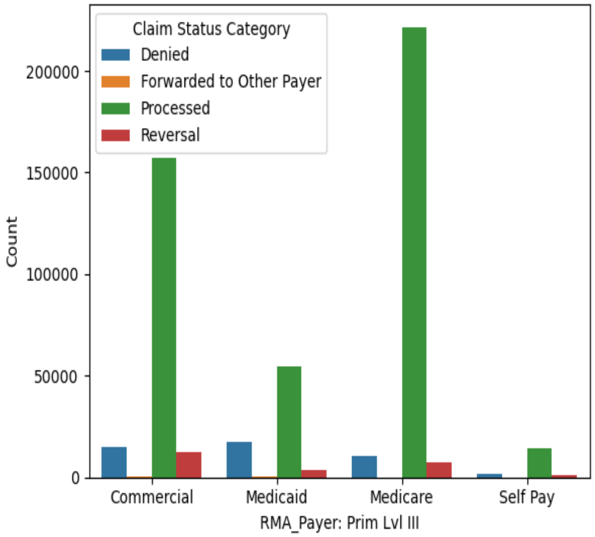
(a)



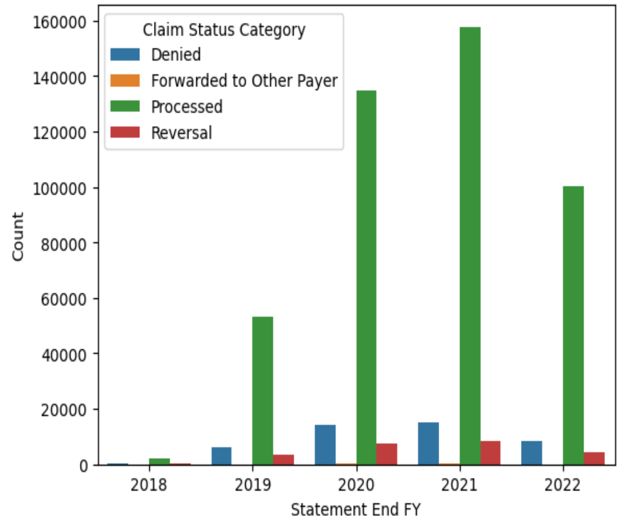
(b)

Fig 2: Scatter plots

(a) Scatter plot of Claims Dataset (b) Scatter plot of remits Data Set

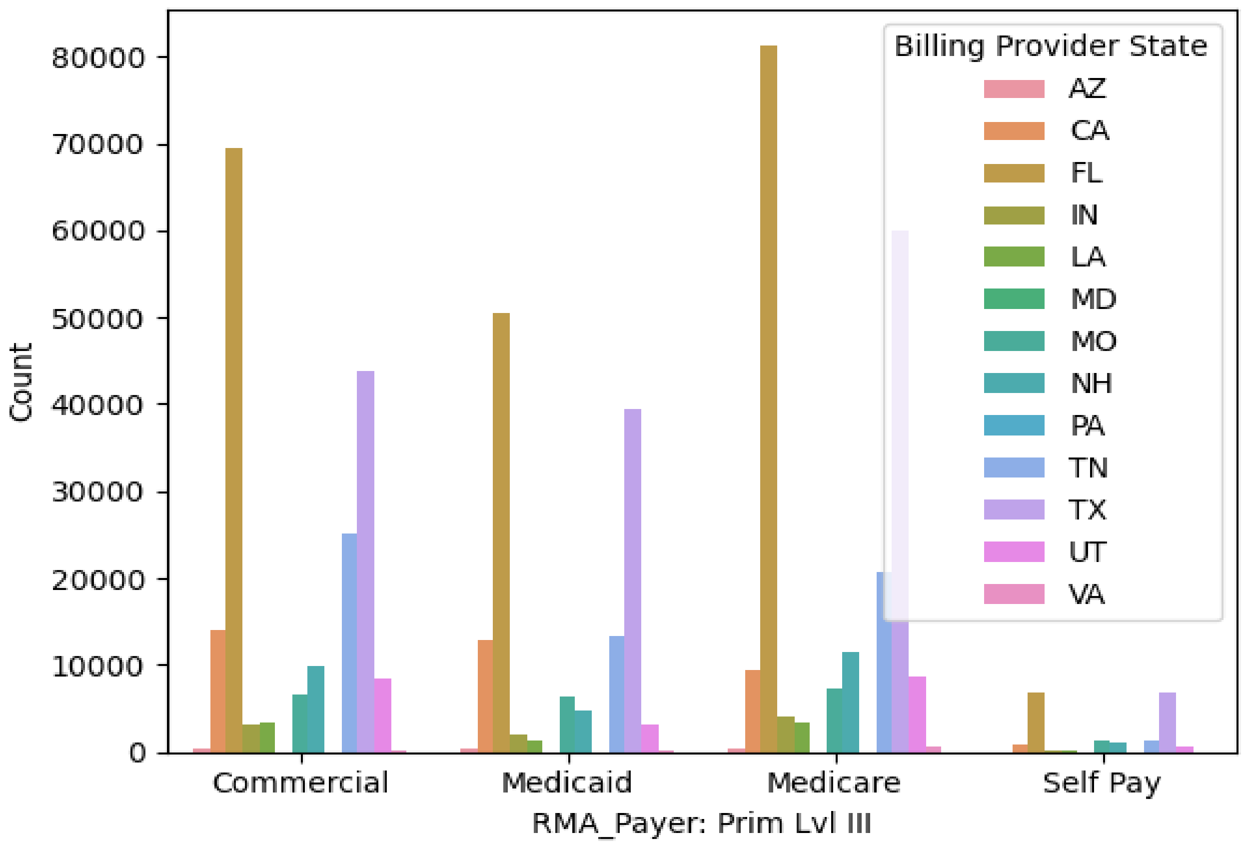


(a)

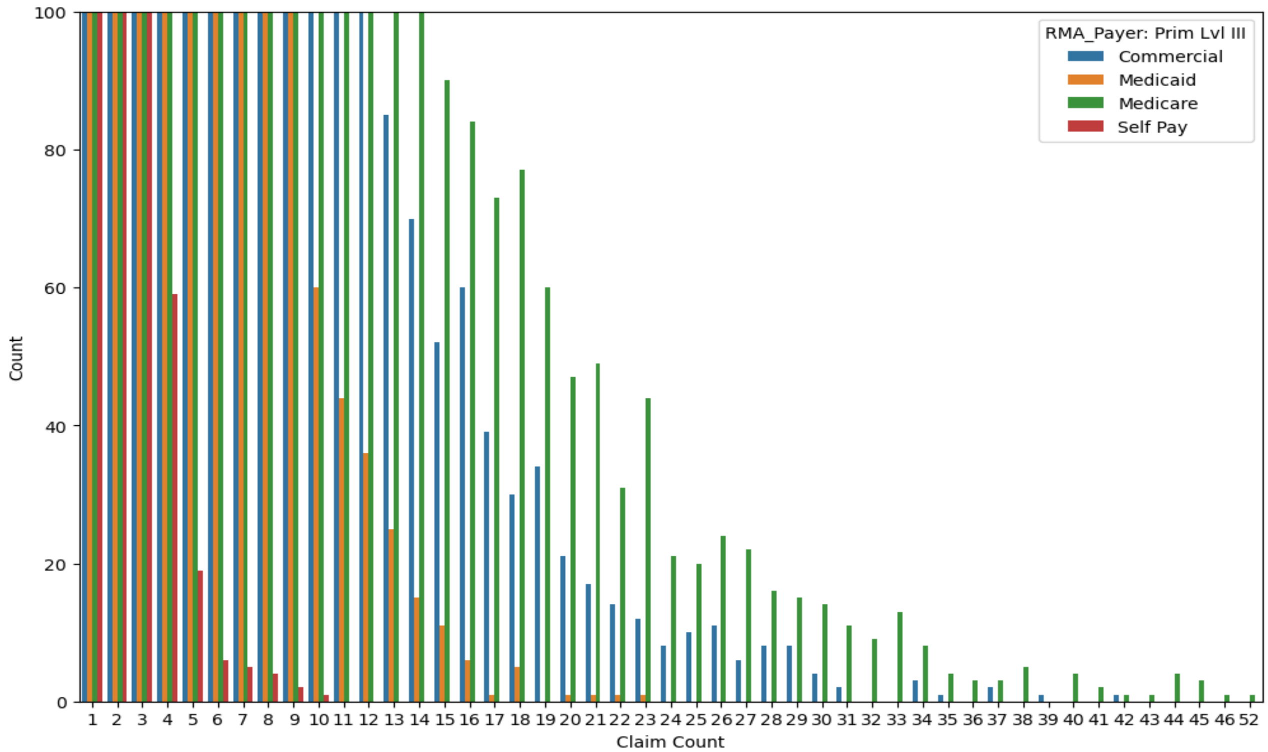


(b)

Fig 3: (a) Stacked bar chart of claim status category and RMA_Payer: Prim Lvl III (b) Stacked Bar Chart of Statement End Year and Claim Status Category



(a)



(b)

Fig 4: (a) Stacked Bar chart of RMA_Payer: Prim Lvl III and Billing Provider State (b) The stacked bar chart of Claim Count and RMA_Payer: Prim Lvl III.

Analysis of time taken for the payment based on the payer type:

From the Multinomial regression it is revealed that there is a change in the log of odds of the outcome and the p-value in the post-covid era compared to its counterpart. During the pre-covid era, the coefficients of Medicaid, Medicare, Self-pay are negative, indicating a decrease in the odds of the outcome. The p-value is more than 0.05, so coefficient is not statistically significant, and the AIC value reveals that the model is not a good model. Also, statistics reveal that the Medicaid claims took less time to get paid, the commercial and Medicare plans took more time. During Post-covid era, the coefficients of Medicaid and Medicare are positive indicating an increase in the odds of the outcome, and the Self-Pay coefficient is negative, indicating a decrease in the odds of the outcome. The p-value is less than 0.05 for the Medicare claims, so the coefficient is statistically significant and the p-value for Self-Pay is less than 0.05, so the coefficient is not significant.

The AIC value reveals that the model is not a good model, and the statistics reveal that the Medicaid Claims took less time to get paid, however Medicaid claims took more time to get paid. All the assumptions of the multinomial regression are met except the linearity assumption for both pre-covid and post-covid dataset. Overall, the Number of days taking for a claim to get paid decreased in the post-covid era, The Medicaid and Medicare insurance types became statistically significant post-covid in regards of the time taken for the payment. So, there is a slight effect of insurance type on the time taken for the payment, the summary is tabulated in Table 3(a) and 3(b).

Table 3 (a) Summary statistics of num_days and RMA_Payer: Prim Lvl III in the pre-COVID era

RMA_Payer: Prim Lvl III	Mean	Minimum	Maximum	Standard Deviation
Commercial	1.245428	0	8172	77.331126
Medicaid	0.763077	0	29	2.460356
Medicare	0.925155	0	7277	56.799696
Self-Pay	0.624544	0	182	7.932618

Table 3 (b) Summary statistics of num_days and RMA_Payer: Prim Lvl III in the post-COVID era

RMA_Payer: Prim Lvl III	Mean	Minimum	Maximum	Standard Deviation
Commercial	0.184567	0	65	1.014510
Medicaid	0.286225	0	28	1.415414
Medicare	0.221591	0	291	1.449940
Self-Pay	0.135210	0	232	2.022633

Analyzing the claim status and number of days needed for claim reimbursement based on state:

From the chi-squared test, it is observed that during the pre-covid and post-covid era, there is a relationship between as the p-value is less than 0.05. The odds ratios for both the eras is 1 revealing that there is a slightly positive association between all the groups of claims status category and Billing Provider State. However, when the standardized residuals is performed to check which cell is contributing towards the chi-squared statistic, it is observed that for denied, processed, and reversal claims during pre-covid era are between -2 and 2 indicating that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic. The standardized residual values for all states except Indiana, Louisiana, Michigan, Tennessee, Utah are less than -2, so they are contributing significantly to overall chi-square test statistic. The standardized residual values during post-covid era for denied, processed, and reversal claims are between -2 and 2 indicating that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic. The standardized residual values for all states except California, Florida, Louisiana, Missouri, New Hampshire, Nevada, South Carolina are less than -2, so are contributing significantly to overall chi-square test statistic.

For the pre-covid era, all the assumptions of the chi-squared test are met except one. Specifically, since all the observations are collected independently, the assumption all observations are independent is met. As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met. The assumption of the expected values should be 5 or higher in at least 80% of groups is not met. However, during the post-covid era, all the assumptions are observed to be met. When the statistics are run, it is observed that during pre-covid era Pennsylvania took more time to complete the payment throughout all status. New

Hampshire, Nevada, and New York took the least time to complete the payment. However, during the post-covid era Indiana took more time to make the payment, Colorado, Iowa, Idaho, Illinois, Maine, Nebraska, Ohio, Oklahoma, South Carolina, Virginia, Puerto Rico took very less time. From the visualizations it is observed that the percentage change in number of days taken to complete the payment is larger in Nevada, and there is no change in South Carolina, Puerto Rico, Oklahoma, Ohio, New York, Nebraska, Manie, Massachusetts, Illinois, Idaho, Iowa, Colorado, and Arizona (Fig 5). Overall, there is a relationship between the Billing Provider State and the Claim Status Category, and the payment time is improved post covid.

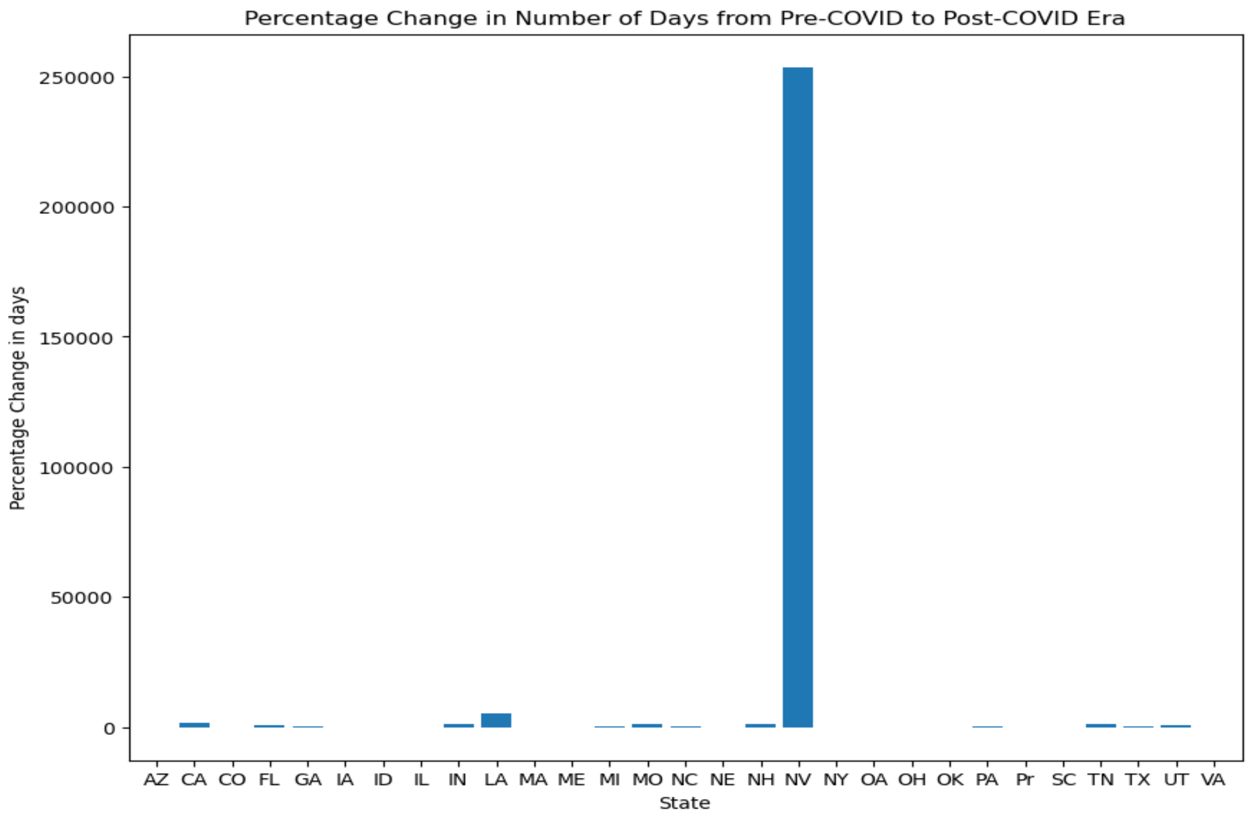


Fig 5: The bar chart of percentage change in number of days from pre-COVID to Post-COVID era

Predictive Analysis

Predicting if the claims are paid or not:

The time taken for fitting the model and the accuracy of logistic regression are 3.42 seconds, and 100 percent respectively. The time taken for fitting the model and the accuracy of Gaussian Naive Bayes are 0.46 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of K-Nearest Neighbors are 87.31 seconds and 99.8 percent respectively. The time taken for fitting the model and the accuracy of Support Vector Machines are 960.21 seconds 99.9 percent respectively. The time taken for fitting the model and the accuracy of Random Forest are 17.33 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of Gradient Boost are 21.85 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of XG Boost are 5.74 seconds and 100 percent respectively. Thus, the best model is Gaussian Naive Bayes as it took less time and provided 100 percent accuracy. The other better models are logistic regression, XG boost, Random Forest, Gradient Boost respectively. The ranking and the summary is given in Table 5.

Table 5: Summary and Ranking of Machine Learning Algorithms

Machine Learning Algorithm	Time in seconds	Accuracy in percentage	Rank
Logistic Regression	3.42	100	2
Gaussian Naive Bayes	0.46	100	1
K-Nearest Neighbors	87.31	99.8	6
Support Vector Machines	960.21	99.9	7
Random Forest	17.33	100	4
Gradient Boost	21.85	100	5
Xg Boost	5.74	100	3

IV. DISCUSSION

The study used the Dask framework in Python and claims and denial data from the Envision Healthcare database to analyze trends in pre-COVID and post-COVID era. The datasets were cleaned by removing nulls and changing data types to appropriate types. Descriptive statistics were calculated for numerical attributes in both datasets, and visualizations like scatter plots, heatmaps, stacked bar charts, and bar charts were used to reveal interactions between variables. Statistical tests like Chi-Squared test and Multinomial regression were used to analyze the datasets, and machine learning classification models like Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost, XG Boost were used to predict if claims were paid or not. A linear regression model was also used to predict the time taken for the payment. The study found a relationship between the type of insurance and claim status category, and that there was a relationship between the type of insurance and the number of days taken for payment. The study also found a relationship between claim status category and billing provider state, with a percentage change in the number of days from pre-COVID to post-COVID. In the predictive analysis it is found that the best model for predicting payment of claims was Random Forest, and the best model for predicting the time taken for payment was Linear Regression.

This study can be used by the organization in a number of ways to improve their operations, increase efficiency, and reduce costs. The organization can use the insights gained from the statistical and predictive analyses to optimize their claims processing procedures. For example, by understanding which types of claims are more likely to be denied, they can proactively identify and resolve issues that might otherwise cause a delay in payment. They can also identify patterns in the data that suggest which types of claims are more likely to be fraudulent, allowing them to prioritize their investigations accordingly. By analyzing the data on the different types of insurance providers and the time it takes for them to pay claims, the organization can better negotiate with payers and set more favorable terms for reimbursement. They can also use the insights to identify payers that are more likely to be problematic, allowing them to focus their attention on resolving issues with those payers. By understanding the patterns in the data related to the time it takes to process claims and receive payment, the organization can better plan their finances and allocate resources more effectively. For example, they can adjust their cash flow projections based on the time it takes to receive payment, reducing the risk of cash flow issues. The visualizations generated in this study can help the organization identify areas where they need to improve their operations. For example, if they see a high percentage of denied claims for a particular insurance provider, they can investigate the cause and take steps to address the issue. Overall, the results of this study can help the organization make data-driven decisions that improve their operations, increase efficiency, and reduce costs.

Though it is beneficial to use the study results to inform organizational decisions, it is important to evaluate the relevance, reliability, practicality, and ethical implications of the findings before taking any action. One of the limitations to the study is that the study results may not be applicable to all situations or populations. The Organization should assess whether the study's findings are relevant to their specific context. The sample size is relatively small as it represents the fraction of the records so, the results might vary with the sample size.

V. CONCLUSION

In conclusion, this study analyzed healthcare claims and remits data to understand the trends and patterns of claim processing and payment. Descriptive statistics and data visualization techniques were used to gain insights into the data, while statistical analyses were performed to test hypotheses and determine the relationships between different variables. Predictive Analysis were performed to predict if the claims will be denied or paid, and the time taken to pay the claims. Overall, this study provides useful insights into the processing and payment of healthcare claims and highlights the need for further research to better understand the factors affecting the time taken for claims reimbursement and the denial of claims by insurance companies. The findings of this study could also be useful for healthcare providers and insurance companies to improve their claim processing and payment systems.

VI. REFERENCES

[1] Kaye AD, Okeagu CN, Pham AD, et al. Economic impact of COVID-19 pandemic on healthcare facilities and systems: International perspectives. *Best Pract Res Clin Anaesthesiol*. 2021;35(3):293-306. doi:10.1016/j.bpa.2020.11.009

[2] Spatharou A, Hieronimus S, Jenkins J. Transforming healthcare with ai: The impact on the workforce and Organizations. McKinsey & Company.

https://www.mckinsey.com/industries/healthcare/our-insights/transforming-healthcare-with-ai. Published March 10, 2020. Accessed April 26, 2023.

[3] J. Wojtusiak, C. Ngufor, J. Shiver and R. Ewald, "Rule-Based Prediction of Medical Claims' Payments: A Method and Initial Application to Medicaid Data," 2011 10th International Conference on Machine Learning and Applications and Workshops, Honolulu, HI, USA, 2011, pp. 162-167, doi: 10.1109/ICMLA.2011.126.

[4] Kumar M, Ghani R, Mei Z-S. Data mining to predict and prevent errors in health insurance claims processing. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2010. doi:10.1145/1835804.1835816

[5] P. Bradley and J. Kaplan, “Turning Hospital Data into Dollars,” Healthcare Financial Management, 64, 2, 2010, 64-68

[6] Li J, Huang K-Y, Jin J, Shi J. A survey on statistical methods for health care fraud detection. *Health Care Management Science*. 2008;11(3):275-287. doi:10.1007/s10729-007-9045-4

[7] Musal RM. Two models to investigate Medicare fraud within unsupervised databases. *Expert Systems with Applications*. 2010;37(12):8628-8633. doi:10.1016/j.eswa.2010.06.095

VII. APPENDIX: PYTHON CODE IN JUPYTER NOTEBOOK

CAPSTONE TITLE

ANALYSIS OF THE EFFECT OF COVID ON THE PAYMENT OF HEALTHCARE EXPENSES AND PAYMENT PREDICTION USING MACHINE LEARNING ALGORITHMS

OBJECTIVE OF CAPSTONE

- The objective of the project is to analyse the payments in pre-covid and post-covid era and predict the claims if they are paid or not and if paid how much time would it take to pay.

BACKGROUND

- The data used in this project is 835 (remits) data and 837 (claims) data.
- The 835 data shows how the claim is paid or denied electronically. When payments are posted with no reconciliation against charges or expected amount
- The 837 data is submitted to an insurance company or clearinghouse instead of sending a paper claim in the mail

PROJECT ROADMAP

- Data Cleaning of both the data sets
- Division of the remits dataset based on the year to filter pre-covid and post-covid era
- statistical analysis to analyse the types of insurances accepting or denying the insurance claim for the payment of healthcare expenses, the time taken for the payment, state wise analysis of the payments Pre-Covid and Post-Covid are analysed.
- Predictice analysis to predict the claims if they are paid or not and if paid how much time would it take to pay.

Importing the necessary libraries

In [1]:

```
import numpy as np
import pandas as pd
import dask.dataframe as ddf
import dask
from dask.delayed import delayed
import pandas as pd
from dask.distributed import Client
from datetime import datetime
from scipy.stats import shapiro, normaltest
import seaborn as sns
```

```
import matplotlib.pyplot as plt
import scipy.stats as ss
import seaborn as sns
from scipy import stats
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from xgboost import XGBClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import dask.dataframe as dd
from dask_ml.model_selection import train_test_split
from dask_ml.preprocessing import OneHotEncoder, StandardScaler
from dask_ml.linear_model import LinearRegression
from dask_ml.wrappers import Incremental
from dask_ml.wrappers import ParallelPostFit
from dask_ml.linear_model import LinearRegression
import dask.array as da
from sklearn.metrics import accuracy_score
import statsmodels.api as sm
import time
```

In [2]:

```
client = Client(n_workers=4)
```

Loading the datasets

Remits Data Set

Information about data:

- Statement End FY -- Year at which the payment is closed either after denial or claimed
- Statement Start Date -- The date when the payment is billed
- Statement End Date -- The date when the payment is payed
- RMA_Payer: Prim Lvl III -- The category of insurance whether it is commercial, medicaid, medicare
- Denial Type -- States whether the payment is denied or not
- Claim Status Description -- description of the claim status category
- Claim Status Category -- The Consolidated naming of Denial type
- Claim Received Date -- The actual date when the payment is received
- Claim Filling Code -- code related to claims
- Billing Provider State -- state where the billing is done
- Billing Provider City -- city where the billing is done
- Initial Denial Remit Count -- the count of the denied claims initially
- Total Denied Remit Count -- the total denied claims count
- Final Denials Remit Count -- the final denied claims count

In [3]:

```
## loading the remits data set
```

```
remits_data = ddf.read_csv(r'835data.csv')
```

In [4]:

```
## checking the data

remits_data.head()
```

Out[4]:

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider State	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count
0	2018	12/31/2018	12/31/2018	Commercial	Not Denied	Processed as Primary	Processed	9/17/2019	12	TX	DALLAS	0	0	0
1	2018	12/31/2018	12/31/2018	Commercial	Not Denied	Processed as Primary	Processed	1/4/2019	13	FL	FORT WALTON BEACH	0	0	0
2	2018	12/31/2018	12/31/2018	Commercial	Not Denied	Reversal of Previous Payment	Reversal	6/27/2019	12	TX	DALLAS	0	0	0
3	2018	12/31/2018	12/31/2018	Commercial	Not Denied	Reversal of Previous Payment	Reversal	1/4/2019	13	FL	FORT WALTON BEACH	0	0	0
4	2018	12/31/2018	12/31/2018	Commercial	Timely Filing	Processed as Primary	Processed	3/2/2021	HM	IN	INDIANAPOLIS	0	1	0

Claims Dataset

Information about the data:

- Statement End FY -- Year at which the payment is closed either after denial or claimed
- Statement Start Date -- The date when the payment is billed
- Statement End Date -- The date when the payment is payed
- RMA_Payer: Prim Lvl III -- The category of insurance whether it is commercial, medicaid, medicare
- Claim Bill Date -- The actual date when the payment is billed
- Billing Provider State -- state where the billing is done
- Billing Provider City -- city where the billing is done
- Claim Count -- the count of the total claims initially

In [5]:

```
## loading the claims data set

claims_data = ddf.read_csv(r'837data.csv')
```

In [6]:

```
## checking the data

claims_data.head()
```

Out[6]:

	Statement End FY	Billing Provider City	Statement End Date	RMA_Payer: Prim Lvl III	Claim Bill Date	Statement Start Date	Admit Type	Billing Provider State	Claim Count
0	2019	ARLINGTON	12/8/2019	Medicaid	8/27/2021	12/8/2019	NaN	TX	1
1	2019	ARLINGTON	11/7/2019	Medicaid	8/27/2021	11/6/2019	NaN	TX	1
2	2019	ARLINGTON	10/23/2019	Medicaid	8/27/2021	10/23/2019	NaN	TX	1
3	2019	ARLINGTON	10/22/2019	Medicaid	8/27/2021	10/22/2019	NaN	TX	1
4	2019	ARLINGTON	9/30/2019	Medicaid	8/27/2021	9/30/2019	NaN	TX	1

EXPLORATORY DATA ANALYSIS

Data Cleaning

Remits Data Set

In [7]:

```
## Dropping Null rows

remits = remits_data.dropna(subset = ["RMA_Payer: Prim Lvl III","Billing
Provider State","Billing Provider City"])
```

In [8]:

```
## changing the datasets as needed

remits["Statement End Date"] = remits["Statement End
Date"].astype("datetime64[ns]")

remits['Claim Status Category'] = remits['Claim Status
Category'].astype('category')

remits['RMA_Payer: Prim Lvl III'] = remits['RMA_Payer: Prim Lvl
III'].astype('category')

remits['Denial Type'] = remits['Denial Type'].astype('category')

remits['Claim Status Description'] = remits['Claim Status
Description'].astype('category')

remits['Claim Status Category'] = remits['Claim Status
Category'].astype('category')

remits['Billing Provider State'] = remits['Billing Provider
State'].astype('category')

remits['Billing Provider City'] = remits['Billing Provider
City'].astype('category')

remits["Statement Start Date"] = remits["Statement Start
Date"].astype("datetime64[ns]")

remits["Claim Received Date"] = remits["Claim Received
Date"].astype("datetime64[ns]")
```

In [9]:

```
## checking the data types

remits.dtypes
```

Out[9]:

```
Statement End FY                      int64
Statement Start Date         datetime64[ns]
Statement End Date           datetime64[ns]
RMA_Payer: Prim Lvl III            category
Denial Type                        category
Claim Status Description           category
Claim Status Category              category
Claim Received Date          datetime64[ns]
Claim Filling Code                   object
Billing Provider State             category
Billing Provider City              category
Initial Denial Remit Count            int64
Total Denied Remit Count              int64
Final Denials Remit Count             int64
```

```
dtype: object
```

In [10]:

```
## creating a variable after calculating the number of days

remits['num_days'] = (remits['Statement End Date'] - remits['Statement Start Date']).dt.days
```

In [11]:

```
## filtering out negative days

remits = remits[remits['num_days'] >= 0]
```

In [12]:

```
# Filter the data for pre-COVID era
remits_pre_covid = remits[remits['Statement Start Date'] < '2020-01-01']


# Filter the data for post-COVID era
remits_post_covid = remits[remits['Statement Start Date'] >= '2020-01-01']
```

Claims Data Set

In [13]:

```
## Dropping null values

claims = claims_data.dropna(subset = ["RMA_Payer: Prim Lvl III"])
```

In [14]:

```
## dropping null column

claims = claims.drop('Admit Type',axis = 1)
```

In [15]:

```
## Changing the data types as needed

claims["Statement End Date"] = claims["Statement End Date"].astype("datetime64[ns]")

claims['Billing Provider City'] = claims['Billing Provider City'].astype('string')

claims['RMA_Payer: Prim Lvl III'] = claims['RMA_Payer: Prim Lvl III'].astype('category')

claims['Billing Provider State'] = claims['Billing Provider State'].astype('category')

claims["Statement Start Date"] = claims["Statement Start Date"].astype("datetime64[ns]")

claims["Claim Bill Date"] = claims["Claim Bill Date"].astype("datetime64[ns]")
```

In [16]:

```
## checking the data types
```

```
claims.dtypes
```

Out[16]:

```
Statement End FY                    int64
Billing Provider City              string
Statement End Date         datetime64[ns]
RMA_Payer: Prim Lvl III          category
Claim Bill Date            datetime64[ns]
Statement Start Date       datetime64[ns]
Billing Provider State           category
Claim Count                         int64
dtype: object
```

Descriptive Statistics

Remits Dataset

In [17]:

```
remits.describe().compute()
```

Out[17]:

	Statement End FY	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
count	516959.000000	516959.000000	516959.000000	516959.000000	516959.000000
mean	2020.657000	0.196577	0.676858	0.016688	0.314354
std	0.968968	0.434827	0.967532	0.136818	21.397328
min	2018.000000	0.000000	0.000000	0.000000	0.000000
25%	2020.000000	0.000000	0.000000	0.000000	0.000000
50%	2021.000000	0.000000	1.000000	0.000000	0.000000
75%	2021.000000	0.000000	1.000000	0.000000	0.000000
max	2022.000000	34.000000	55.000000	28.000000	8172.000000

INTERPRETATION: There are 516964 total records. Initially 34 remits are denied at maximum and 28 the final denial remit count. However most of the claim count are received after the third quartile (75%). The maximum total remits count is 55. The maximum number of days needed to complete the payment of claims is 8172.

Claims Dataset

In [18]:

```
claims.describe().compute()
```

Out[18]:

	Statement End FY	Claim Count
count	544606.000000	544606.00000
mean	2020.549895	1.38842
std	0.939618	1.40679
min	2019.000000	1.00000
25%	2020.000000	1.00000
50%	2021.000000	1.00000
75%	2021.000000	1.00000
max	2022.000000	52.00000

INTERPRETATION: The maximum number of claims received from insurance companies is 52 claims.

Data Visualization

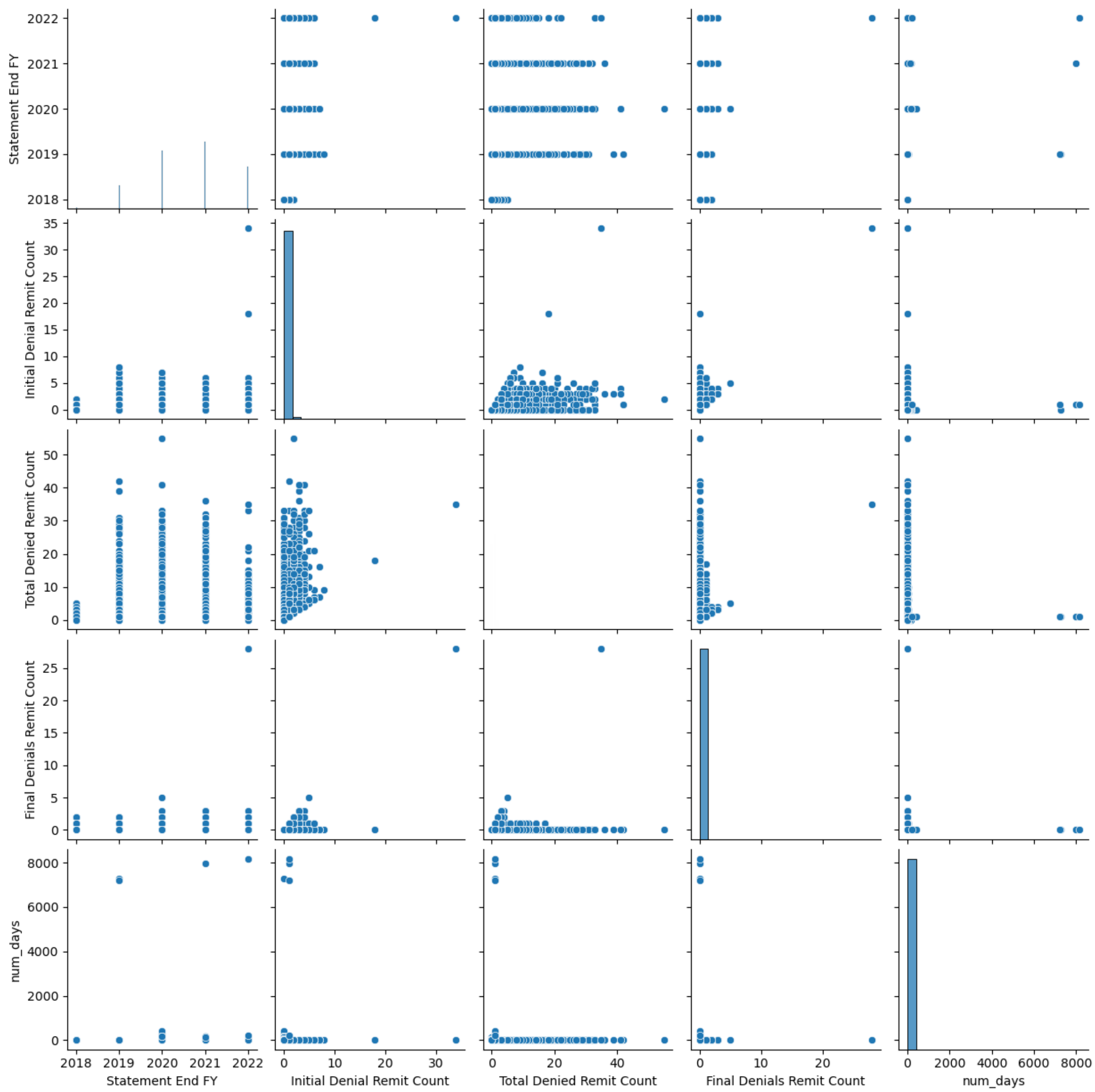
Remits Dataset

In [19]:

```
## plotting pair plot to check correlation

sns.pairplot(remits.compute())
```

Out[19]:

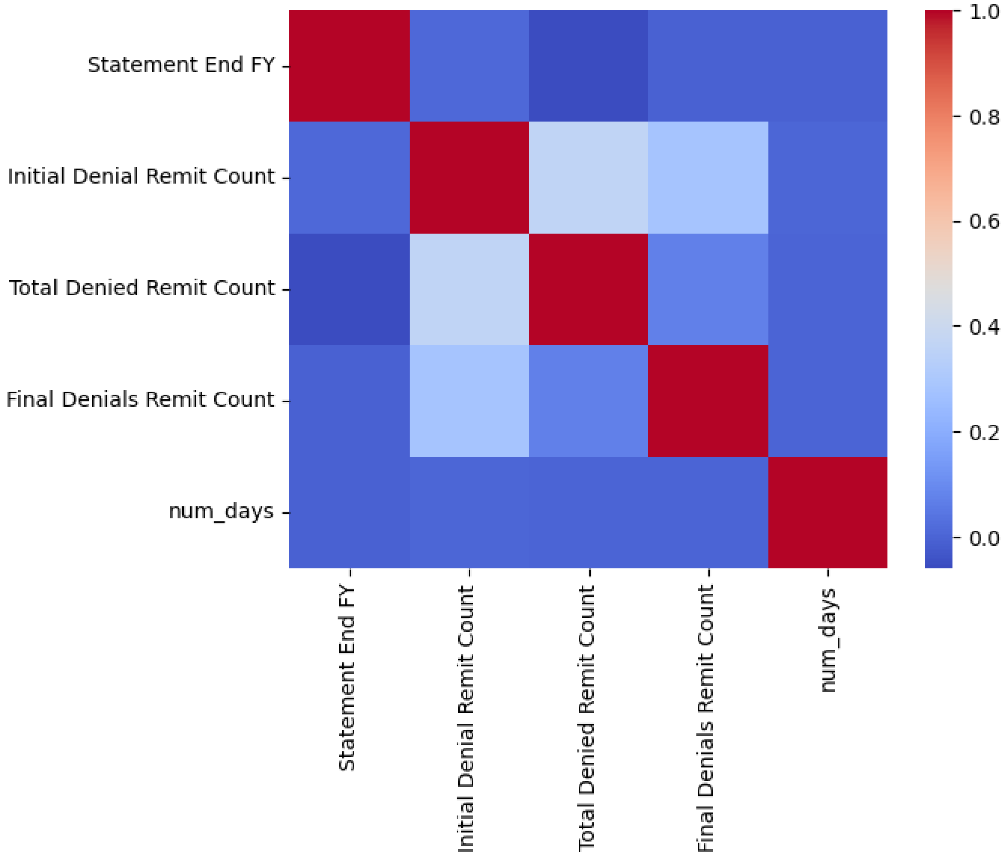
```
<seaborn.axisgrid.PairGrid at 0x13db1662bb0>
```



Interpretation: There is no linearity between number of days needed for the payment and the denial remit count

In [20]:

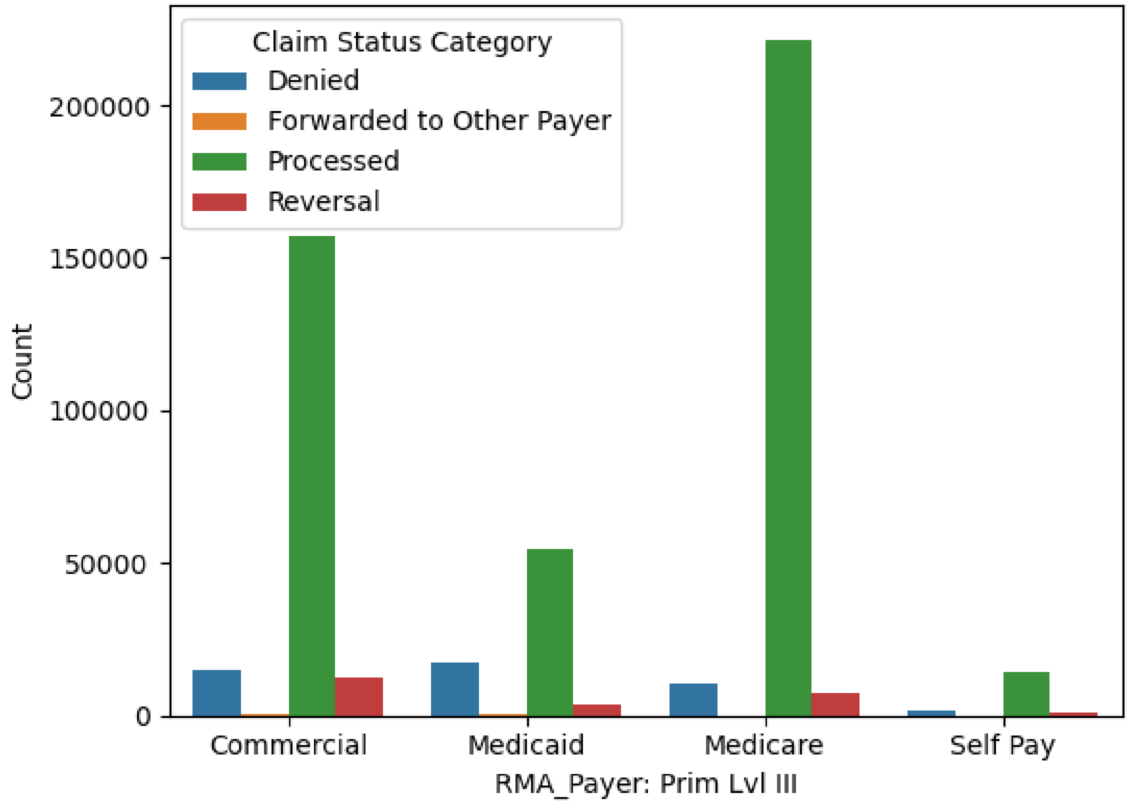
```
## plotting correlation map

corr = remits.corr().compute()
sns.heatmap(corr, cmap='coolwarm')
plt.show()
```



INTERPRETATION: As expected there is a correlation between the denial remit count. There is also a slight corelation between year and initial denial remit count

In [119]:

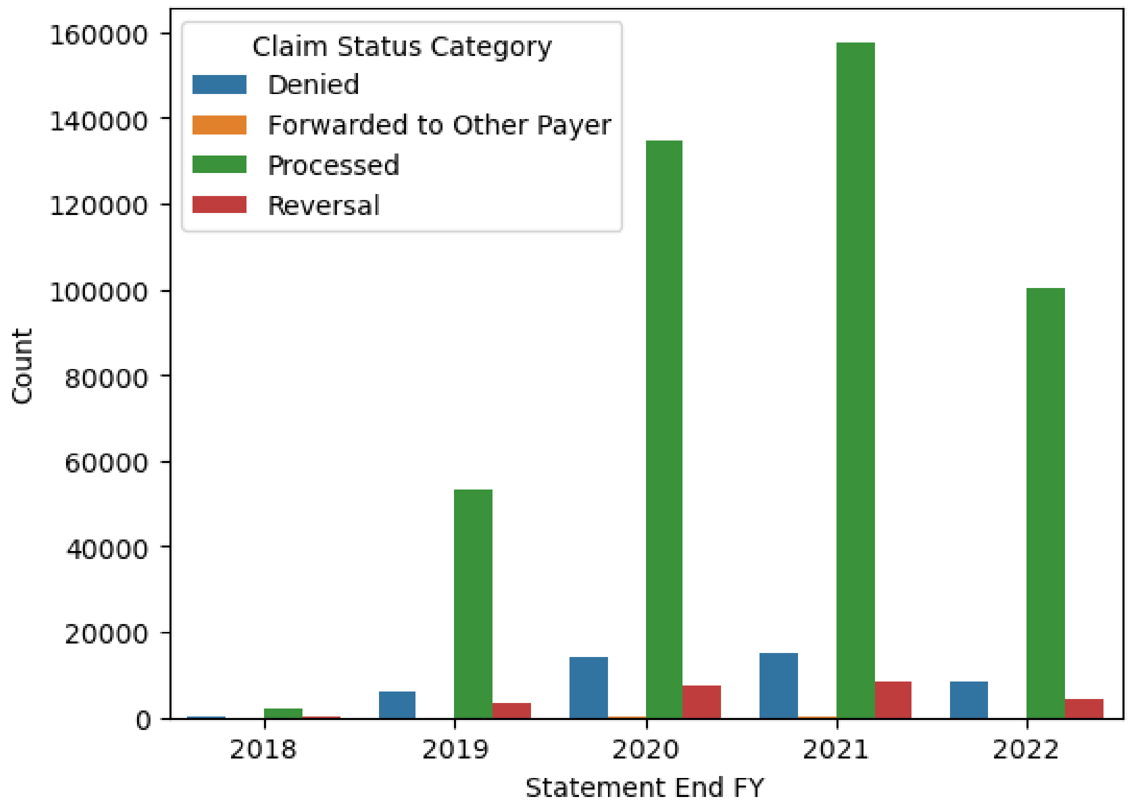
```
## stacked bar chart between claim status category

# Group data by two categorical variables
grouped = remits.groupby(['RMA_Payer: Prim Lvl III', 'Claim Status
Category']).size().reset_index().compute()
grouped
# Plot bar chart
sns.barplot(x='RMA_Payer: Prim Lvl III', y=0, hue='Claim Status Category',
data=grouped)
plt.ylabel('Count')
plt.show()
```



INTERPRETATION: Most of the Medicare payments are preocessed i.e., the claims are retrieved. Among all the payers, Medicaid insurances are the most denied ones. Among all the payers the commercial payers had more claims which are reversed. None or very few claims are forworded to other payer among all types payers.

In [120]:

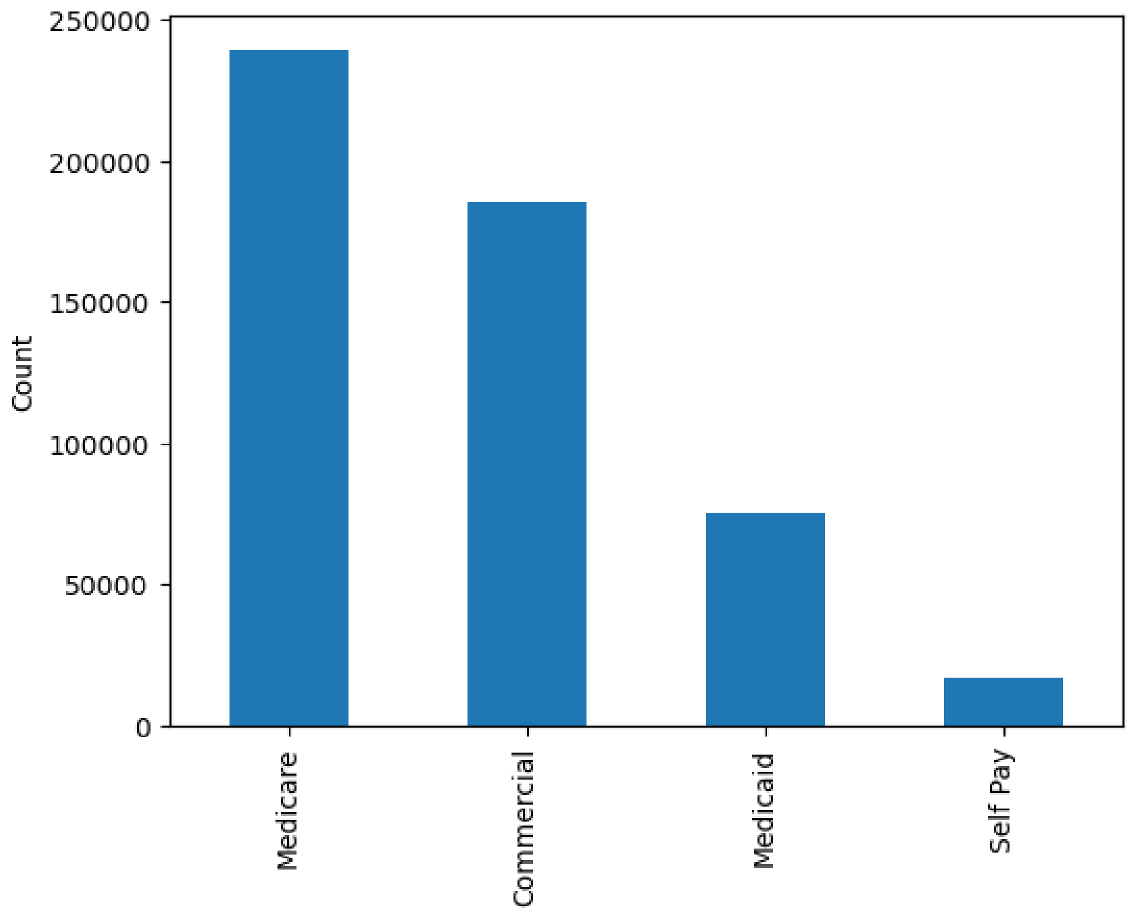
```
## stacked bar chart between year and claim status category

# Group data by two categorical variables
grouped = remits.groupby(['Statement End FY', 'Claim Status
Category']).size().reset_index().compute()
grouped
# Plot bar chart
sns.barplot(x='Statement End FY', y=0, hue='Claim Status Category',
data=grouped)
plt.ylabel('Count')
plt.show()
```



INTERPRETATION: Most of the processed claims are in the year 2021 and the least are in 2018. Most of the claims are denied in the year 2021. Very few claims are deied in 2018.

In [121]:

```
## Bar chart for outcome variable

counts = remits['RMA_Payer: Prim Lvl III'].value_counts().compute()
counts.plot(kind='bar')
plt.ylabel('Count')
plt.show()
```

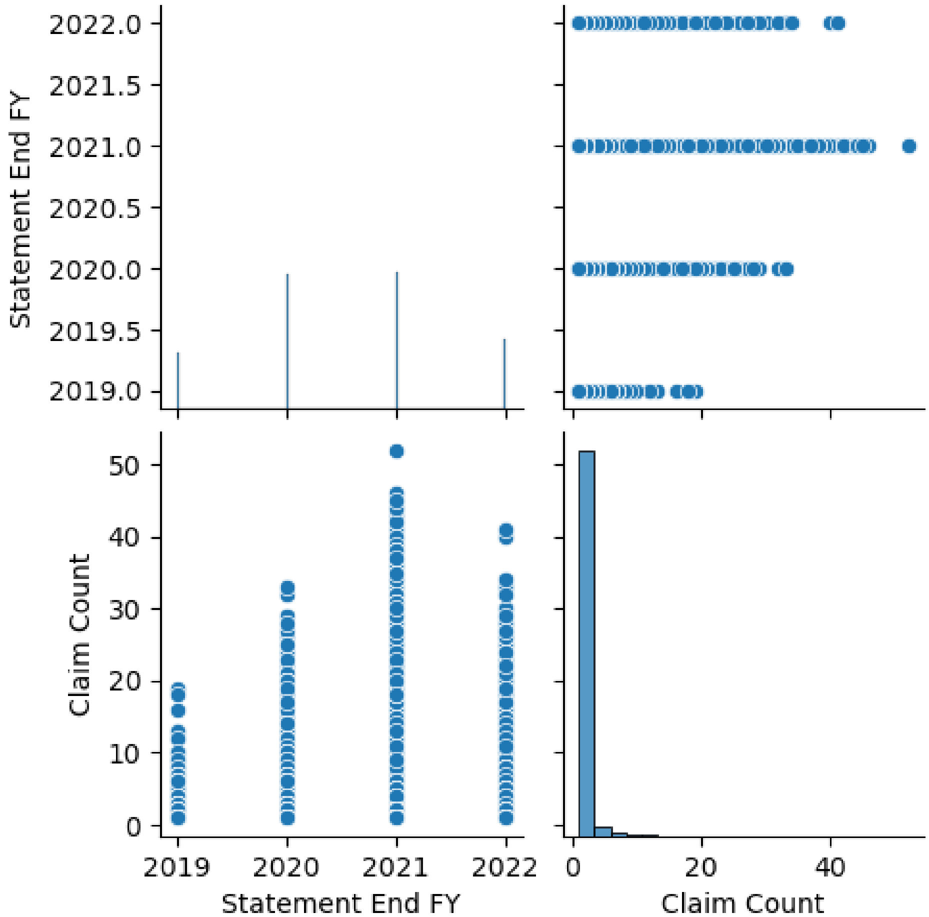


INTERPRETATION: Most of the records in the data set are medicaid payers and very few are from self pay.

Claims Dataset

In [24]:

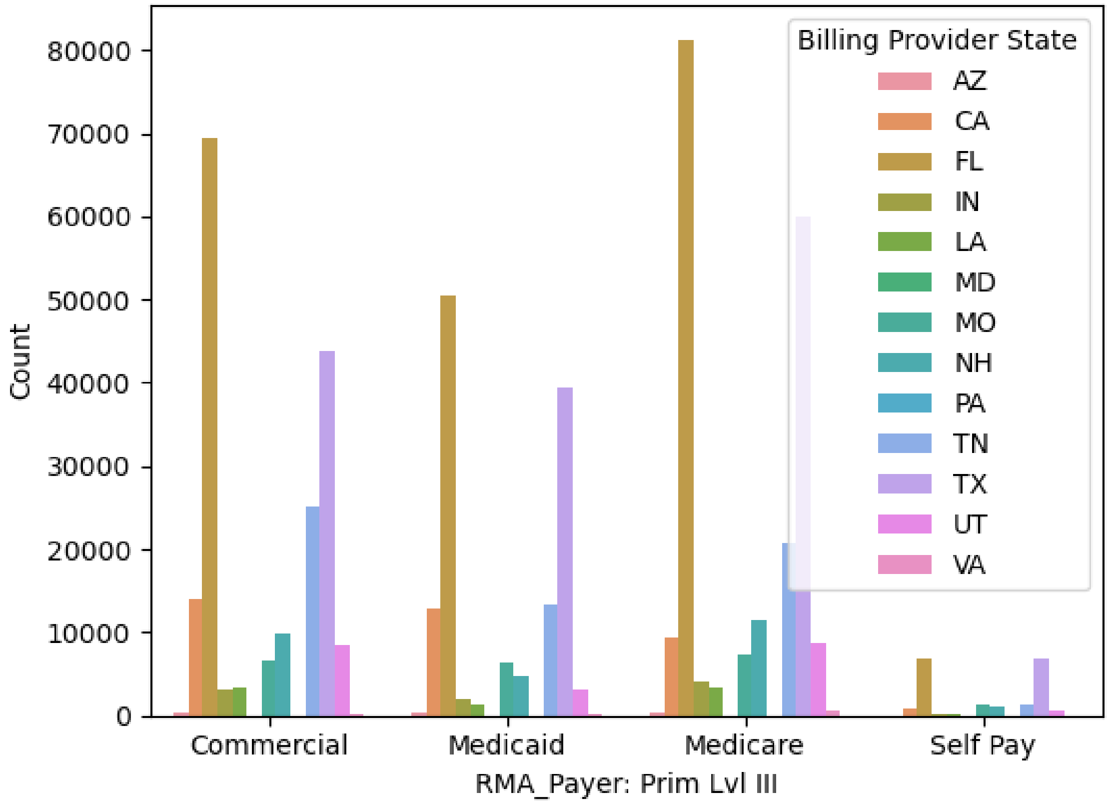
```
## Pair plot

sns.pairplot(claims.compute())
plt.show()
```



INTERPRETATION: There is no linearity between the claim count and year.

In [122]:

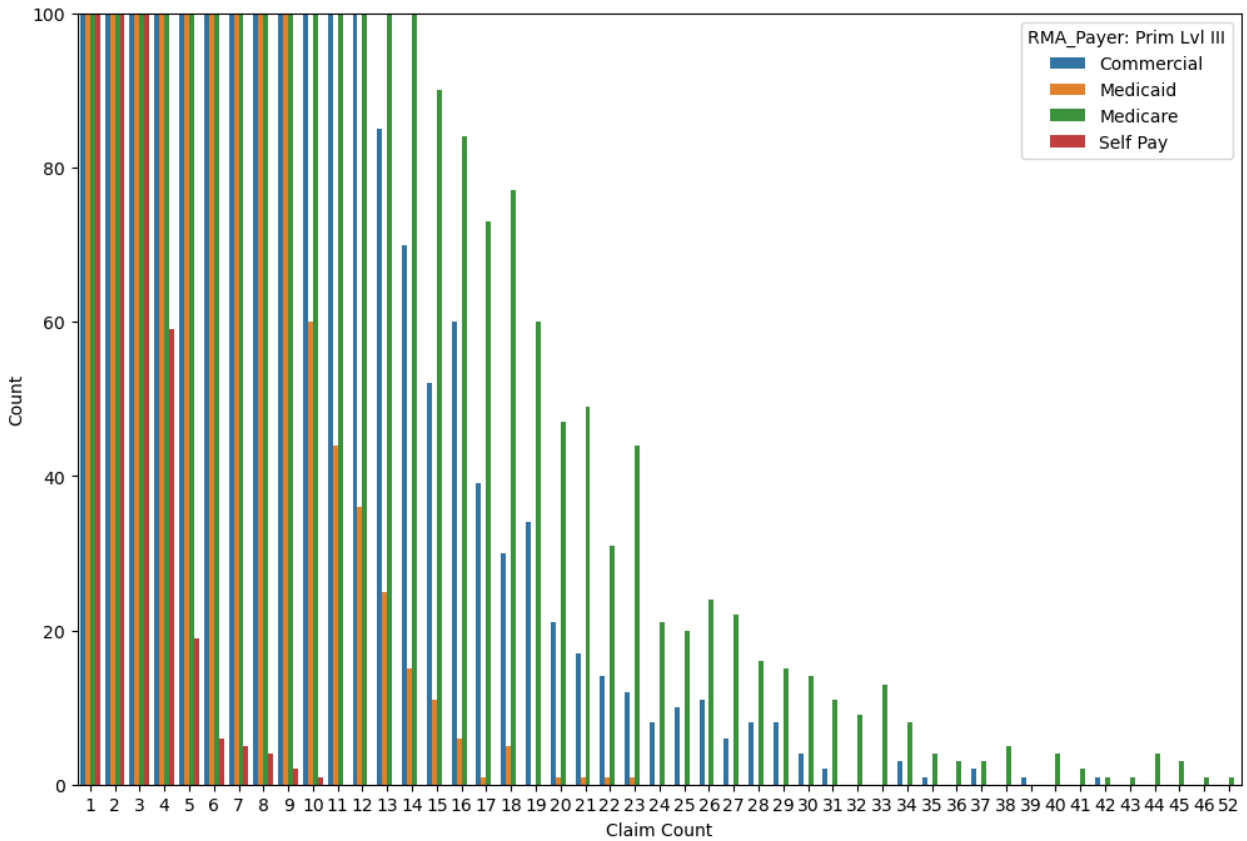
```
## stacked bar chart between Payer Level III and State

# Group data by two categorical variables
grouped = claims.groupby(['RMA_Payer: Prim Lvl III', 'Billing Provider State'
]).size().reset_index().compute()
grouped
# Plot bar chart
sns.barplot(x='RMA_Payer: Prim Lvl III', y=0, hue='Billing Provider State' ,
data=grouped)
plt.ylabel('Count')
plt.show()
```



INTERPRETATION: Among all others states, Florida has most records in all the payer types and among those Medicare has most records. Other states having most records in all payers is Texas and Tennessee.

In [123]:

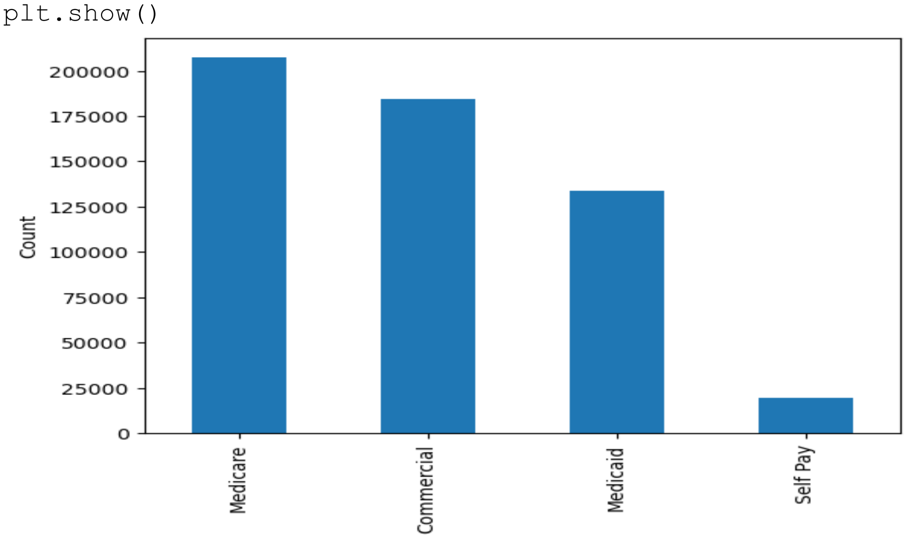
```
## stacked bar chart between claim count and Payer Lvl III

# Group data by two categorical variables
grouped = claims.groupby(['RMA_Payer: Prim Lvl III','Claim
Count']).size().reset_index().compute()
grouped
# Plot bar chart
fig, ax = plt.subplots(figsize=(12, 8))
ax = sns.barplot(x='Claim Count', y=0, hue='RMA_Payer: Prim Lvl III',
data=grouped)
ax.set(ylim=(0, 100)) # set the y-axis limits
plt.ylabel('Count')
plt.show()
2023-05-07 18:25:44,659 - distributed.utils_perf - WARNING - full garbage
collections took 19% CPU time recently (threshold: 10%)
```



INTERPRETATION: Most of the payers had a count below 10 claims.

In [124]:

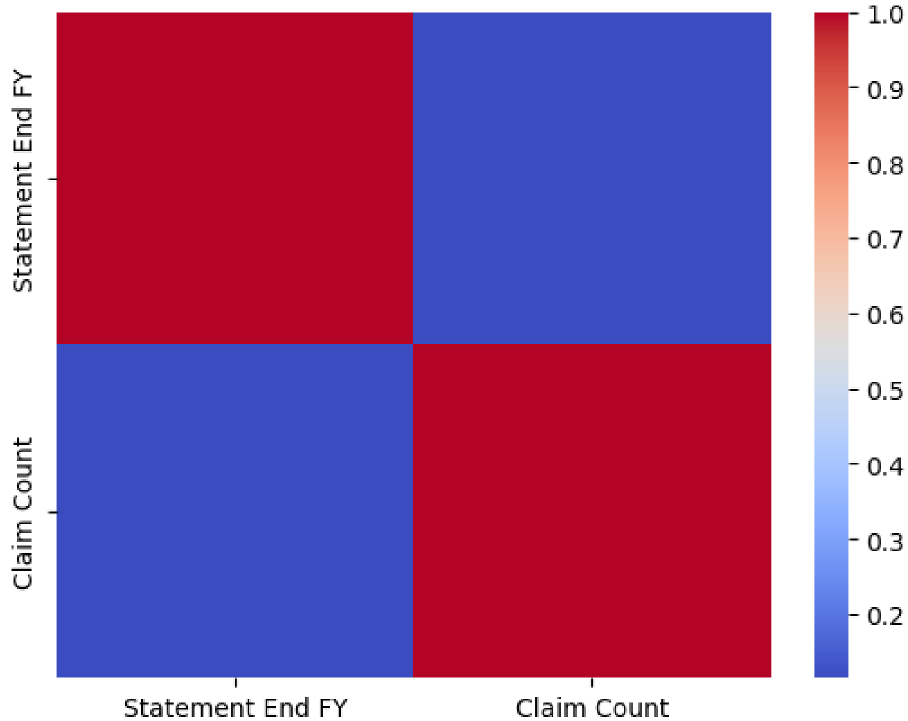
```
## bar chart for knowing the count of Payer Lvl III

counts = claims['RMA_Payer: Prim Lvl III'].value_counts().compute()
counts.plot(kind='bar')
plt.ylabel('Count')
plt.show()
```



INTERPRETATION: In claims dataset too medicare had more records and less self pay.

In [28]:

```
## correlation map

corr = claims.corr().compute()
sns.heatmap(corr, cmap='coolwarm')
plt.show()
```



INTERPRETATION:There is no correlation between the claim count and year.

STATISTICAL ANALYSIS

Research question 1: Analyzing the types of insurances accepting or denying the insurance claim for the payment of healthcare expenses in pre-covid and post covid era

- To analyse the acceptance of insurances Statistical tests are performed along with the assumption testing. After that proportions and percentages are calculated to know which insurances are denied and accepted.
- Statistical tests used in this analysis is Chi Squared test as we are analysing the Claim status of the various types of insurances if they are paid or not. The outcome variable in this analysis is RMA_Payer:Prim Lvl III and the predictor variable is Claim Status Category and both the variables are categorical.

Pre-covid era

Hypothesis

Null Hypothesis: There is no relationship between Claim Status Category and RMA_Payer: prim Lvl III

Alternate Hypothesis: There is a relationship between Claim Status Category and RMA_Payer: Prim Lvl III

In [29]:

```
observed = remits_pre_covid.groupby(['Claim Status Category', 'RMA_Payer:
Prim Lvl III']).size().compute().unstack()

chi2, pval, dof, expected = ss.chi2_contingency(observed)

print(f"Chi-squared: {chi2:.2f}")
print(f"p-value: {pval:.2f}")
print(f"Degrees of freedom: {dof}")
Chi-squared: 8341.46
p-value: 0.00
Degrees of freedom: 9
```

INTERPRETATION: Since the P-Value is less than 0.05, we reject null hypothesis. So we can conclude that there is a relationship between the RMA_Payer: prim Lvl III and Claim Status Category

In [30]:

```
# calculate the standardized residuals
standardized_residuals = (observed - expected) / expected.std()

print(standardized_residuals)
RMA_Payer: Prim Lvl III   Commercial  Medicaid  Medicare  Self Pay
Claim Status Category
Denied                     -0.103392  0.296111 -0.216103  0.023385
Forwarded to Other Payer    0.001898 -0.000232 -0.001579 -0.000086
Processed                   0.029532 -0.317251  0.319811 -0.032092
Reversal                    0.071962  0.021372 -0.102128  0.008794
```

INTERPRETATION: Since standardized residual value between -2 and 2 indicates that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic.

In [31]:

```
# convert counts to proportions
prop_tab = observed.div(observed.sum(axis=1), axis=0)

# calculate odds ratios
ref_category = remits['Claim Status Category']
odds_ratios = prop_tab.div(prop_tab.loc[ref_category])

# display the odds ratios
print(odds_ratios)
RMA_Payer: Prim Lvl III  Commercial  Medicaid  Medicare  Self Pay
Claim Status Category
Denied                          1.0       1.0       1.0       1.0
Denied                          1.0       1.0       1.0       1.0
```

```
Denied                          1.0       1.0       1.0       1.0
Denied                          1.0       1.0       1.0       1.0
Denied                          1.0       1.0       1.0       1.0
...                             ...       ...       ...       ...
Reversal                        1.0       1.0       1.0       1.0
Reversal                        1.0       1.0       1.0       1.0
Reversal                        1.0       1.0       1.0       1.0
Reversal                        1.0       1.0       1.0       1.0
Reversal                        1.0       1.0       1.0       1.0

[516959 rows x 4 columns]
```

INTERPRETATION: Since the odds ratios is 1 there is slightly a positive association between all the groups of claims status category and RMA_Payer: Prim Lvl III

Assumptions of chi-squared: 1) All observations should be independent. 2) All the variable should be nominal or ordinal 3) The expected values should be 5 or higher in at least 80% of groups.

In [32]:

```
## Checking whether the expected values should be 5 or higher in 80% of
groups

grouped_denial = remits_pre_covid.groupby('Claim Status Category')

# Calculate the expected values for each group
expected_denial = grouped_denial.mean()

# Check the assumption that 80% of groups have expected values of 5 or higher
num_above_threshold = (expected_denial >= 5).sum().compute()
num_groups = len(expected_denial)
percent_above_threshold = num_above_threshold / num_groups * 100
percent_above_threshold
if (percent_above_threshold >= 80.0).any():
    print("Assumption met!")
else:
    print("Assumption not met.")
Assumption met!
```

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.
- As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met.
- The assumption of The expected values should be 5 or higher in at least 80% of groups is met

In [33]:

```
denied_claims_grouped = remits_pre_covid.groupby(['Claim Status Category'])
denied_claims_grouped.count().compute()
```

Out[33]:

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Received Date	Claim Filling Code	Billing Provider State	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Claim Status Category														
Denied	6460	6460	6460	6460	6460	6460	6460	6460	6460	6460	6460	6460	6460	6460
Forwarded to Other Payer	26	26	26	26	26	26	26	26	26	26	26	26	26	26
Processed	55292	55292	55292	55292	55292	55292	55292	55292	55292	55292	55292	55292	55292	55292
Reversal	3764	3764	3764	3764	3764	3764	3764	3764	3764	3764	3764	3764	3764	3764

In [34]:

```
# Group data by insurance provider
grouped = remits_pre_covid.groupby('RMA_Payer: Prim Lvl III')

# Calculate the proportion of accepted and denied claims for each provider
results = grouped['Claim Status Category'].value_counts().compute()
```

In [35]:

```
results
```

Out[35]:

```
RMA_Payer: Prim Lvl III
Commercial                Denied                        1369
                          Forwarded to Other Payer        23
                          Processed                    18630
                          Reversal                      1797
Medicaid                  Denied                        3171
                          Forwarded to Other Payer         2
                          Processed                     5585
                          Reversal                       705
Medicare                  Denied                        1581
                          Forwarded to Other Payer         1
```

```
                          Processed                   29931
                          Reversal                     1101
Self Pay                  Denied                        339
                          Forwarded to Other Payer        0
                          Processed                    1146
                          Reversal                      161
Name: Claim Status Category, dtype: int64
```

In [36]:

```
# group the data by insurance type and claim status
grouped = remits_pre_covid.groupby(['RMA_Payer: Prim Lvl III', 'Claim Status
Category'])

# count the number of claims in each category
counts = grouped.size()

# calculate the total number of claims for each insurance type
totals = counts.groupby('RMA_Payer: Prim Lvl III').sum()

# calculate the percentage of accepted and denied claims for each insurance
type
percentages = (counts / totals * 100).compute()

# view the resulting percentages
print(percentages)
RMA_Payer: Prim Lvl III  Claim Status Category
Commercial               Denied                       6.274348
                         Forwarded to Other Payer     0.105413
                         Processed                   85.384298
                         Reversal                     8.235941
Medicaid                 Denied                      33.509458
                         Forwarded to Other Payer     0.021135
                         Processed                   59.019338
                         Reversal                     7.450069
Medicare                 Denied                       4.847611
                         Forwarded to Other Payer     0.003066
                         Processed                   91.773472
                         Reversal                     3.375851
Self Pay                 Denied                      20.595383
                         Forwarded to Other Payer     0.000000
                         Processed                   69.623329
                         Reversal                     9.781288
dtype: float64
```

INTERPRETATION: Out of the total 21819 commercial payers 18630 claims are processed which constitutes of 85.3% and only 6.17 % of claims are denied. Among medicaid claims 33.5% of them are denied and only 59% of them are processed. Among all the 32614 medicare claims 91.77% of them are processed and just 4% of them are denied. Finally 20% of the self pay claims are denied.

Post-covid era

Hypothesis

Null Hypothesis: There is no relationship between Claim Status Category and RMA_Payer: prim Lvl III

Alternate Hypothesis: There is a relationship between Claim Status Category and RMA_Payer: Prim Lvl III

In [37]:

```
# perform a chi-squared test on two categorical variables
contingency_table = pd.crosstab(remits_post_covid['Claim Status Category'],
remits_post_covid['RMA_Payer: Prim Lvl III'])
chi2, p, dof, expected = ss.chi2_contingency(contingency_table)
```

In [38]:

```
print(f"Chi-squared: {chi2:.2f}")
print(f"p-value: {p:.2f}")
print(f"Degrees of freedom: {dof}")
Chi-squared: 22286.96
p-value: 0.00
Degrees of freedom: 9
```

INTERPRETATION: Since the P-Value is less than 0.05, the null hypothesis is rejected. Thus we can conclude that there is relationship between Claim Status Category and RMA_Payer: Prim Lvl III

In [39]:

```
# calculate the standardized residuals
standardized_residuals = (contingency_table - expected) / expected.std()

print(standardized_residuals)Out of the total 21819 commercial payers 18630
claims are processed which constituthes of 85.3% and only 6.17 % of claims
are denied. Among medicaid claims 33.5% of them are denied and only 59% of
them are processed. Among all the 32614 medicare claims 91.77% of them are
processed and just 4% of them are denied. Finally 20% of the self pay claims
are denied.
col_0                      Commercial  Medicaid  Medicare  Self Pay
row_0
Denied                      -0.002621  0.159887 -0.162822  0.005555
Forwarded to Other Payer     0.001735  0.001375 -0.002931 -0.000179
Processed                   -0.064264 -0.162920  0.231138 -0.003954
Reversal                     0.065150  0.001658 -0.065386 -0.001422
```

INTERPRETATION: Since standardized residual value between -2 and 2 indicates that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic.

In [40]:

```
# convert counts to proportions
prop_tab = contingency_table.div(contingency_table.sum(axis=1), axis=0)

# calculate odds ratios
ref_category = remits['Claim Status Category']
odds_ratios = prop_tab.div(prop_tab.loc[ref_category])

# display the odds ratios
print(odds_ratios)
col_0     Commercial  Medicaid  Medicare  Self Pay
row_0
Denied           1.0       1.0       1.0       1.0
```

```
Denied          1.0     1.0     1.0     1.0
Denied          1.0     1.0     1.0     1.0
Denied          1.0     1.0     1.0     1.0
Denied          1.0     1.0     1.0     1.0
...             ...     ...     ...     ...
Reversal        1.0     1.0     1.0     1.0
Reversal        1.0     1.0     1.0     1.0
Reversal        1.0     1.0     1.0     1.0
Reversal        1.0     1.0     1.0     1.0
Reversal        1.0     1.0     1.0     1.0

[516959 rows x 4 columns]
```

INTERPRETATION: Since the odds ratios is 1 there is slightly a positive association between all the groups of claims status category and RMA_Payer: Prim Lvl III

Assumptions of chi-squared: 1) All observations should be independent. 2) All the variable should be nominal or ordinal 3) The expected values should be 5 or higher in at least 80% of groups.

In [41]:

```
grouped_denial = remits_post_covid.groupby('Claim Status Category')

# Calculate the expected values for each group
expected_denial = grouped_denial.mean()

# Check the assumption that 80% of groups have expected values of 5 or higher
num_above_threshold = (expected_denial >= 5).sum().compute()
num_groups = len(expected_denial)
percent_above_threshold = num_above_threshold / num_groups * 100
percent_above_threshold
if (percent_above_threshold >= 80.0).any():
    print("Assumption met!")
else:
    print("Assumption not met.")
Assumption met!
```

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.
- As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met.
- The assumption of The expected values should be 5 or higher in at least 80% of groups is met

In [42]:

```
denied_claims_grouped = remits_post_covid.groupby(['Claim Status Category'])
denied_claims_grouped.count().compute()
```

Out[42]:

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Received Date	Claim Filling Code	Billing Provider State	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Claim Status Category														
Denied	37834	37834	37834	37834	37834	37834	37834	37834	37834	37834	37834	37834	37834	37834
Forwarded to Other Payer	362	362	362	362	362	362	362	362	362	362	362	362	362	362
Processed	392841	392841	392841	392841	392841	392841	392841	392841	392841	392841	392841	392841	392841	392841
Reversal	20380	20380	20380	20380	20380	20380	20380	20380	20380	20380	20380	20380	20380	20380

In [43]:

```
# Group data by insurance provider
grouped = remits_post_covid.groupby('RMA_Payer: Prim Lvl III')

# Calculate the proportion of accepted and denied claims for each provider
results = grouped['Claim Status Category'].value_counts().compute()
```

In [44]:

```
results
```

Out[44]:

```
RMA_Payer: Prim Lvl III
Commercial                 Denied                      13561
                           Forwarded to Other Payer      222
                           Processed                  138866
                           Reversal                    10793
Medicaid                   Denied                      13913
                           Forwarded to Other Payer      125
                           Processed                   48926
                           Reversal                     3068
Medicare                   Denied                       8774
                           Forwarded to Other Payer       12
```

```
                            Processed                      191811
                            Reversal                         5896
Self Pay                    Denied                           1586
                            Forwarded to Other Payer            3
                            Processed                       13238
                            Reversal                          623
Name: Claim Status Category, dtype: int64
```

In [45]:

```
# group the data by insurance type and claim status
grouped = remits_post_covid.groupby(['RMA_Payer: Prim Lvl III', 'Claim Status
Category'])

# count the number of claims in each category
counts = grouped.size()

# calculate the total number of claims for each insurance type
totals = counts.groupby('RMA_Payer: Prim Lvl III').sum()

# calculate the percentage of accepted and denied claims for each insurance
type
percentages = (counts / totals * 100).compute()

# view the resulting percentages
print(percentages)
RMA_Payer: Prim Lvl III  Claim Status Category
Commercial               Denied                       8.297133
                         Forwarded to Other Payer     0.135828
                         Processed                   84.963473
                         Reversal                     6.603566
Medicaid                 Denied                      21.070087
                         Forwarded to Other Payer     0.189302
                         Processed                   74.094378
                         Reversal                     4.646232
Medicare                 Denied                       4.249054
                         Forwarded to Other Payer     0.005811
                         Processed                   92.889832
                         Reversal                     2.855303
Self Pay                 Denied                      10.265372
                         Forwarded to Other Payer     0.019417
                         Processed                   85.682848
                         Reversal                     4.032362
dtype: float64
```

INTERPRETATION: Out of the total 163443 commercial payers 138867 claims are processed which constitutes of 84.9% and only 8.29 % of claims are denied. Among medicaid claims 21% of them are denied and 74% of them are processed. Among all the medicare claims 92.88% of them are processed and just 4.24% of them are denied. Finally 10% of the self pay claims are denied.

Report: Chi-square test on both pre-covid (2018-2019) and post covid data (2020-2022) reveals that the payer types and the claim status category were related with each other. And it is observed that the denial of commercial claims increased by 2% but the self pay denial decreased by

10%. Intrestingly the payment trend was positive during the post-covid period.

Research question 2: Analyzing the relationship between the payer and type time taken for the receving the payment

- To analyse the time taken for the payment Statistical tests are performed along with the assumption testing.
- Statistical tests used in this analysis is logistic regression as we are analysing the number of days taken for the statement to end and various types of insurances. The outcome variable used in this analysis is RMA_Payer : Prim Lvl III and the predictor variable is Number of days. Both the outcome variable is quantitative and the predictor variable is categorical.

Pre-covid era

In [46]:

```
# Create dummy variables for the categorical predictor
predictor = remits_pre_covid['num_days']

# Fit the model
model = sm.MNLogit(remits_pre_covid['RMA_Payer: Prim Lvl III'],
sm.add_constant(predictor)).fit()

# Print the summary statistics
print(model.summary())

print("AIC:", model.aic)

from sklearn.metrics import accuracy_score

## Calculate the accuracy score
X_test = sm.add_constant(remits_pre_covid['num_days'])
y_pred = model.predict(X_test)
accuracy = accuracy_score(remits_pre_covid['RMA_Payer: Prim Lvl III'],
y_pred.argmax(axis=1))
print("Accuracy:", accuracy)

# Perform the likelihood ratio test
#print('Likelihood ratio test:')
#print(model.compare_lr_test(model))
Optimization terminated successfully.
         Current function value: 1.085407
         Iterations 8
                          MNLogit Regression Results
==============================================================================
=
Dep. Variable:                      y   No. Observations:
65542
Model:                        MNLogit   Df Residuals:
65536
```

```
Method:                           MLE   Df Model:                           
3
Date:                Sat, 06 May 2023   Pseudo R-squ.:               4.551e-
06
Time:                        22:22:53   Log-Likelihood:                  -
71140.
converged:                       True   LL-Null:                         -
71140.
Covariance Type:            nonrobust   LLR p-value:                      
0.8855
==============================================================================
=
y=Medicaid       coef    std err          z      P>|z|      [0.025      
0.975]
------------------------------------------------------------------------------
-
const         -0.8353      0.012    -67.847      0.000      -0.859      -
0.811
x1            -0.0001      0.000     -0.563      0.573      -0.001       
0.000
------------------------------------------------------------------------------
-
y=Medicare       coef    std err          z      P>|z|      [0.025      
0.975]
------------------------------------------------------------------------------
-
const          0.4020      0.009     45.962      0.000       0.385       
0.419
x1         -7.764e-05      0.000     -0.570      0.569      -0.000       
0.000
------------------------------------------------------------------------------
-
y=Self Pay       coef    std err          z      P>|z|      [0.025      
0.975]
------------------------------------------------------------------------------
-
const         -2.5842      0.026   -101.073      0.000      -2.634      -
2.534
x1            -0.0003      0.001     -0.287      0.774      -0.002       
0.002
==============================================================================
=
AIC: 142291.4537775805
Accuracy: 0.0
```

INTERPRETATION: Here the commercial is used as a base category. Since the coefficients of Medicaid, Medicare, Self-pay are negative, this indicates a decrease in the odds of the outcome. As the p-value is more than 0.05, so coefficient is not statistically significant. The AIC value reveals that the model is not a good model.

In [47]:

```
# Compute summary statistics of num_days
stats_pre = remits_pre_covid.groupby('RMA_Payer: Prim Lvl
III')['num_days'].agg(['mean', 'min', 'max', 'std'])
stats_pre.compute()
```

Out[47]:

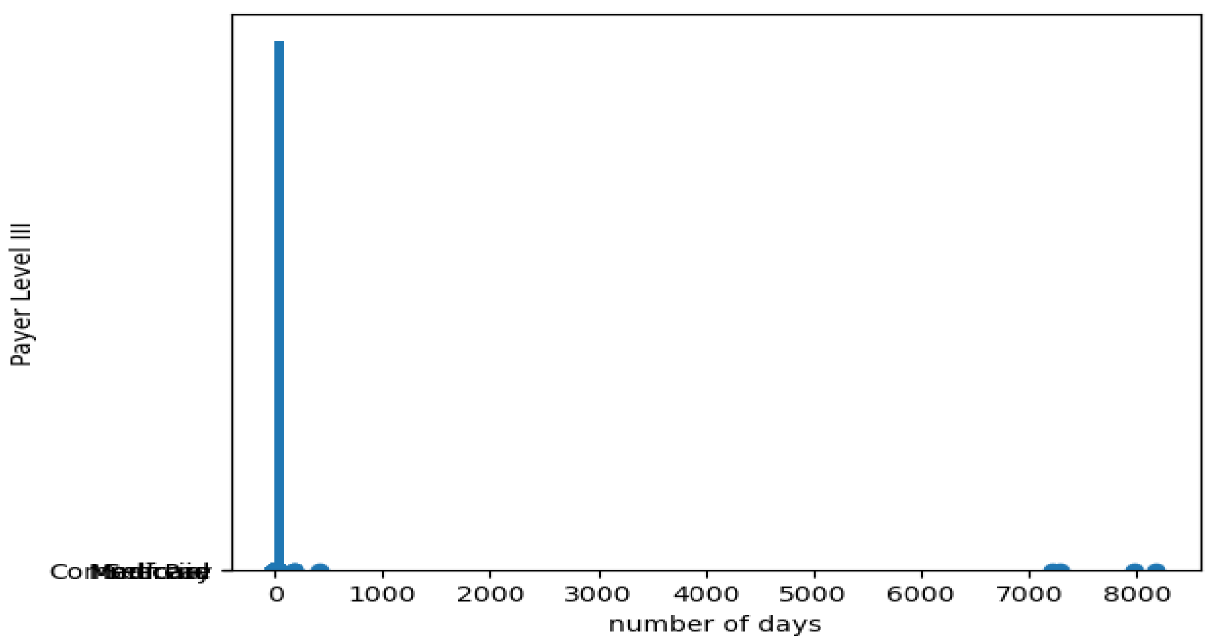
	mean	min	max	std
RMA_Payer: Prim Lvl III				
Commercial	1.245428	0	8172	77.331126
Medicaid	0.763077	0	29	2.460356
Medicare	0.925155	0	7277	56.799696
Self Pay	0.624544	0	182	7.932618

INTERPRETATION: The medicaid claims took less time to get paid, the commercial and Medicare plans took more time.

Assumptions of Mutlinomial Regression: 1) Independence of observations 2) Lineraity 3) No perfect multicollinearity

In [50]:

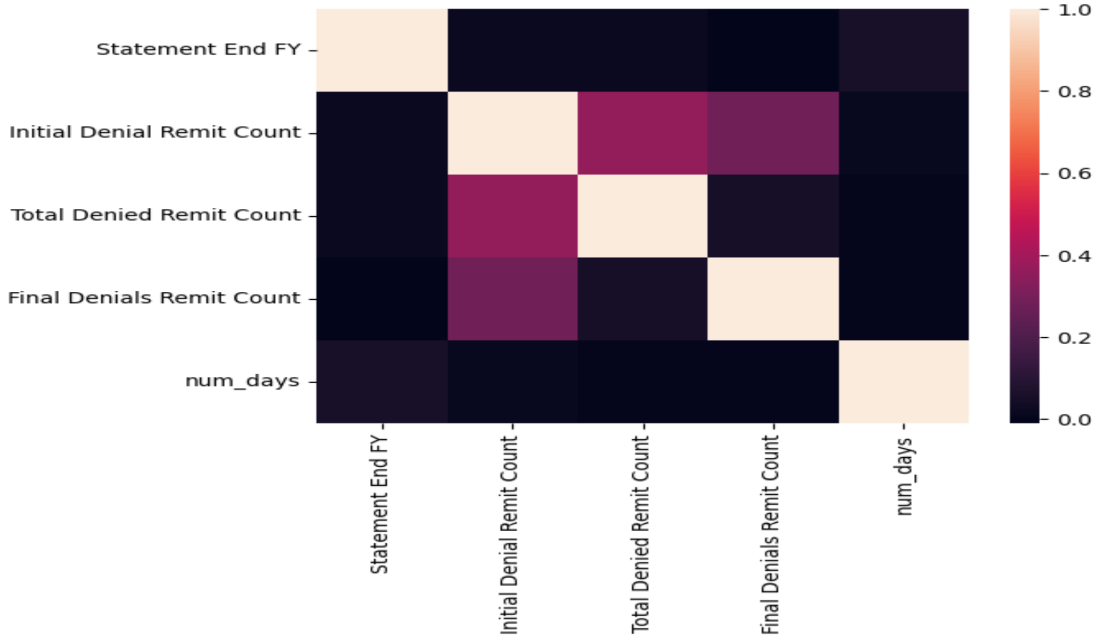
```
# create scatter plot for variable_1 vs. dependent variable
plt.scatter(remits_pre_covid['num_days'].compute(),
remits_pre_covid['RMA_Payer: Prim Lvl III'].compute())
plt.xlabel('number of days')
plt.ylabel('Payer Level III')
plt.show()
```



In [51]:

```
## Plotting correlation matrix to check multicollinearity
```

```
corr = remits_pre_covid.corr().compute()
sns.heatmap(corr)
plt.show()
```



In [52]:

```
corr
```

Out[52]:

	Statement End FY	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Statement End FY	1.000000	0.016098	0.017001	-0.009746	0.055398
Initial Denial Remit Count	0.016098	1.000000	0.364706	0.279340	0.010064
Total Denied Remit Count	0.017001	0.364706	1.000000	0.052565	0.001883
Final Denials Remit Count	-0.009746	0.279340	0.052565	1.000000	-0.000847
num_days	0.055398	0.010064	0.001883	-0.000847	1.000000

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.

- The linearity assumption is not met.
- The correlatin plot reveals that there is no perfect multicollinearity, so this assumption is met.

Post-covid era

In [53]:

```
predictor = remits_post_covid['num_days']
# Fit the model
model = sm.MNLogit(remits_post_covid['RMA_Payer: Prim Lvl III'],
sm.add_constant(predictor)).fit()

# Print the summary statistics
print(model.summary())
print("AIC:", model.aic)

from sklearn.metrics import accuracy_score

## Calculate the accuracy score
X_test = sm.add_constant(remits_post_covid['num_days'])
y_pred = model.predict(X_test)
accuracy = accuracy_score(remits_post_covid['RMA_Payer: Prim Lvl III'],
y_pred.argmax(axis=1))
print("Accuracy:", accuracy)
Optimization terminated successfully.
         Current function value: 1.121820
         Iterations 8
                          MNLogit Regression Results
============================================================================
=
Dep. Variable:                      y   No. Observations:
451417
Model:                        MNLogit   Df Residuals:
451411
Method:                           MLE   Df Model:
3
Date:                Sat, 06 May 2023   Pseudo R-squ.:
0.0004171
Time:                        22:23:57   Log-Likelihood:            -
5.0641e+05
converged:                       True   LL-Null:                   -
5.0662e+05
Covariance Type:            nonrobust   LLR p-value:                 2.764e-
91
============================================================================
=
y=Medicaid       coef    std err          z      P>|z|      [0.025
0.975]
----------------------------------------------------------------------------
-
const         -0.9212      0.005   -195.743      0.000      -0.930      -
0.912
x1             0.0674      0.004     16.394      0.000       0.059
0.075
----------------------------------------------------------------------------
-
```

```
y=Medicare        coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
-
const           0.2260      0.003     66.854      0.000       0.219
0.233
x1              0.0385      0.003     11.095      0.000       0.032
0.045
------------------------------------------------------------------------------
-
y=Self Pay        coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
-
const          -2.3450      0.009   -272.550      0.000      -2.362      -
2.328
x1             -0.0878      0.013     -6.789      0.000      -0.113      -
0.062
==============================================================================
=
AIC: 1012829.4705483892
Accuracy: 0.0
```

INTERPRETATION: Here the commercial is used as a base category. Since the coefficients of Medicaid, Medicare, are positive There is an increase in the odds of the outcome, and since Self-Pay coeeficient is negative, this indicates a decrease in the odds of the outcome. As the p-value is less than 0.05 for the Medicare claims, so coefficient is statistically significant. The p-value for Self-Pay is less than 0.05, so the coeffient is not significant. The AIC value reveals that the model is not a good model.

In [54]:

```
# Compute summary statistics of num_days
stats_post = remits_post_covid.groupby('RMA_Payer: Prim Lvl
III')['num_days'].agg(['mean', 'min', 'max', 'std'])
stats_post.compute()
```

Out[54]:

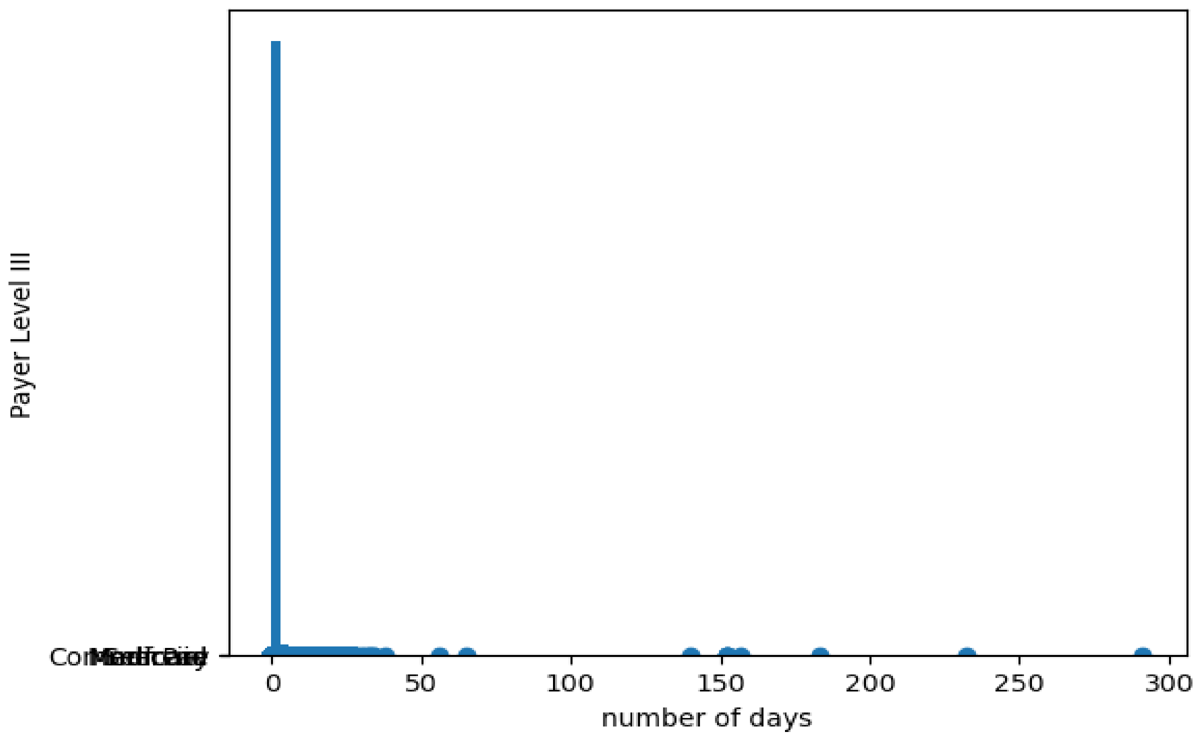
	mean	min	max	std
RMA_Payer: Prim Lvl III				
Commercial	0.184567	0	65	1.014510
Medicaid	0.286225	0	28	1.415414
Medicare	0.221591	0	291	1.449940
Self Pay	0.135210	0	232	2.022633

INTERPRETATION: The medicaid Claims took less time to get paid, however Medicaid claims took more time to get paid.

Assumptions of Mutlinomial Regression: 1) Independence of observations 2) Linerity 3) No perfect multicollinearity

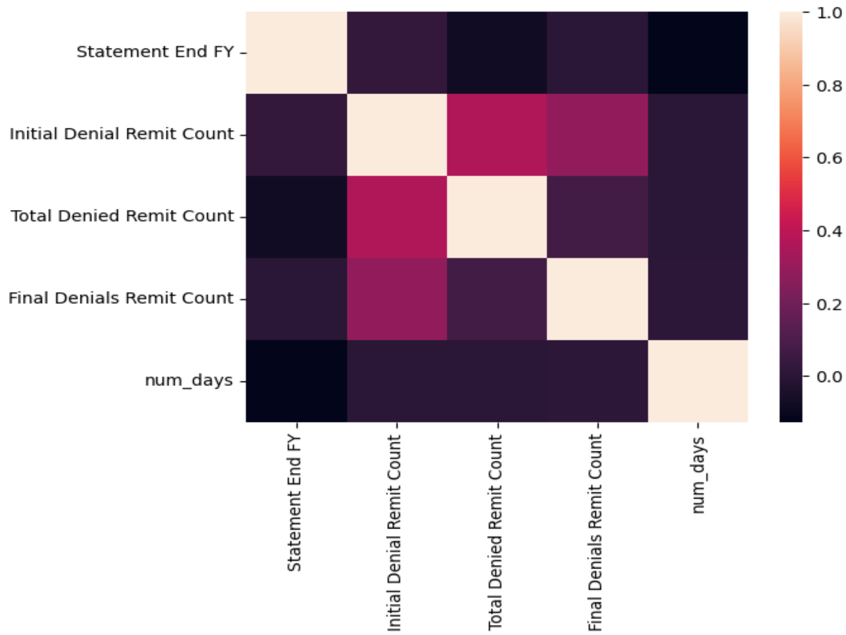
In [56]:

```
# create scatter plot for variable_1 vs. dependent variable to check
linearity

plt.scatter(remits_post_covid['num_days'].compute(),
remits_post_covid['RMA_Payer: Prim Lvl III'].compute())
plt.xlabel('number of days')
plt.ylabel('Payer Level III')
plt.show()
```



In [57]:

```
## creating correlation plot to know if there is multicollinearity

corr = remits_post_covid.corr().compute()
sns.heatmap(corr)
plt.show()
```



In [58]:

```
corr
```

Out[58]:

	Statement End FY	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Statement End FY	1.000000	0.026108	-0.076022	0.000668	-0.125925
Initial Denial Remit Count	0.026108	1.000000	0.366025	0.283281	-0.002052
Total Denied Remit Count	-0.076022	0.366025	1.000000	0.074122	0.000678
Final Denials Remit Count	0.000668	0.283281	0.074122	1.000000	0.006189
num_days	-0.125925	-0.002052	0.000678	0.006189	1.000000

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.
- The linearity assumption is not met.
- The correlation plot reveals that there is no perfect multicollinearity, so this assumption is met.

Report: The Number of days taking for a claim to get paid decreased in the post-covid era, The Medicaid and Medicare insurance types became statistically significant post-covid in regards of the time taken for the payment. So there is a slight effect of insurance type on the time taken for the payment.

Research question 3: Analyzing the claim status and number of days needed for claim reimbursement based on state

To analyse the claim status based on state statistical tests along with visualization of percent change in claim status by state. One statistical test Chi-square test is performed as we have two variables and we are finding the relationship between the outcome variable Billing Provider State in this analysis and predictor variables Claim Status Category (categorical variable).

Pre-covid Era

Chi-Squared Test between Claim Status Category and Billing Provider State

Hypothesis

Null Hypothesis: There is no relationship between Claim Status Category and Billing Provider State

Alternate Hypothesis: There is a relationship between Claim Status Category and Billing Provider State

In [59]:

```
# perform a chi-squared test on two categorical variables
contingency_table_pre = pd.crosstab(remits_pre_covid['Billing Provider
State'], remits_pre_covid['Claim Status Category'])
chi2, p, dof, expected = ss.chi2_contingency(contingency_table_pre)

print(f"Chi-squared: {chi2:.2f}")
print(f"p-value: {p:.2f}")
print(f"Degrees of freedom: {dof}")
Chi-squared: 2714.45
p-value: 0.00
Degrees of freedom: 45
```

INTERPRETATION: Since the P-value is less than 0.05, we reject null hypothesis. So, there is a relationship between billing provider state and claim status category.

In [60]:

```
# calculate the standardized residuals
standardized_residuals = (contingency_table_pre - expected) / expected.std()

print(standardized_residuals)
```

```
col_0    Denied  Forwarded to Other Payer  Processed  Reversal
row_0
CA     0.007883             -2.857278e-05  -0.006887 -0.000967
FL    -0.109250             -5.173959e-04   0.082103  0.027664
GA    -0.000114             -4.571645e-07   0.000180 -0.000066
IN    -0.101653             -1.393701e-03   0.149964 -0.046917
LA    -0.000256             -1.028620e-06  -0.000459  0.000715
MI    -0.038904             -1.565788e-04   0.040120 -0.001060
MO    -0.009144             -3.680174e-05   0.006153  0.003027
NC     0.091162             -7.257486e-05  -0.080871 -0.010218
NH    -0.010734             -4.320204e-05   0.009541  0.001237
NV    -0.000085             -3.428734e-07  -0.000153  0.000238
NY    -0.000085             -3.428734e-07   0.000135 -0.000050
OA    -0.000057             -2.285822e-07   0.000090 -0.000033
PA     0.057773              4.262180e-05  -0.055232 -0.002584
TN    -0.004251             -1.942949e-05   0.004779 -0.000508
TX     0.117819              2.247007e-03  -0.148875  0.028810
UT    -0.000104             -1.897233e-05  -0.000588  0.000711
```

INTERPRETATION: Since standardized residual values for denied, processed, and reversal claims are between -2 and 2 indicating that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic. The standardized residual values for all states except Indiana, Lousiana, Michigan, Tennessee, Utah are less than -2, so they are contributing significantly to overall chi-square test statistic.

In [61]:

```
# convert counts to proportions
prop_tab = contingency_table_pre.div(contingency_table_pre.sum(axis=1),
axis=0)

# calculate odds ratios
ref_category = remits_pre_covid['Billing Provider State']
odds_ratios = prop_tab.div(prop_tab.loc[ref_category])

# display the odds ratios
print(odds_ratios)
col_0  Denied  Forwarded to Other Payer  Processed  Reversal
row_0
CA        1.0                       NaN        1.0       1.0
CA        1.0                       NaN        1.0       1.0
CA        1.0                       NaN        1.0       1.0
CA        1.0                       NaN        1.0       1.0
CA        1.0                       NaN        1.0       1.0
...       ...                       ...        ...       ...
UT        1.0                       NaN        1.0       1.0
UT        1.0                       NaN        1.0       1.0
UT        1.0                       NaN        1.0       1.0
UT        1.0                       NaN        1.0       1.0
UT        1.0                       NaN        1.0       1.0

[65542 rows x 4 columns]
```

INTERPRETATION: Since the odds ratios is 1 there is slightly a positive association between all the groups of claims status category and Billing Provider State

Assumptions of chi-squared: 1) All observations should be independent. 2) All the variable should be nominal or ordinal 3) The expected values should be 5 or higher in at least 80% of groups.

In [62]:

```
grouped_denial = remits_pre_covid.groupby('Billing Provider State')

# Calculate the expected values for each group
expected_denial = grouped_denial.mean()

# Check the assumption that 80% of groups have expected values of 5 or higher
num_above_threshold = (expected_denial >= 5).sum().compute()
num_groups = len(expected_denial)
percent_above_threshold = num_above_threshold / num_groups * 100
percent_above_threshold
if (percent_above_threshold >= 80.0).any():
    print("Assumption met!")
else:
    print("Assumption not met.")
Assumption not met.
```

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.
- As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met.
- The assumption of The expected values should be 5 or higher in at least 80% of groups is not met

In [70]:

```
denied_claims_grouped = remits_pre_covid.groupby(['Billing Provider State'])
denied_claims_grouped.count().compute()
```

Out[70]:

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
AZ	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
CA	250	250	250	250	250	250	250	250	250	250	250	250	250	250
CO	0	0	0	0	0	0	0	0	0	0	0	0	0	0
FL	4527	4527	4527	4527	4527	4527	4527	4527	4527	4527	4527	4527	4527	4527
GA	4	4	4	4	4	4	4	4	4	4	4	4	4	4
IA	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ID	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IL	0	0	0	0	0	0	0	0	0	0	0	0	0	0
IN	17236	17236	17236	17236	17236	17236	17236	17236	17236	17236	17236	17236	17236	17236
LA	9	9	9	9	9	9	9	9	9	9	9	9	9	9
MA	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ME	0	0	0	0	0	0	0	0	0	0	0	0	0	0
MI	1370	1370	1370	1370	1370	1370	1370	1370	1370	1370	1370	1370	1370	1370
MO	322	322	322	322	322	322	322	322	322	322	322	322	322	322
NC	635	635	635	635	635	635	635	635	635	635	635	635	635	635
NE	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
NH	378	378	378	378	378	378	378	378	378	378	378	378	378	378
NV	3	3	3	3	3	3	3	3	3	3	3	3	3	3
NY	3	3	3	3	3	3	3	3	3	3	3	3	3	3
OA	2	2	2	2	2	2	2	2	2	2	2	2	2	2
OH	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OK	0	0	0	0	0	0	0	0	0	0	0	0	0	0
PA	17273	17273	17273	17273	17273	17273	17273	17273	17273	17273	17273	17273	17273	17273
Pr	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SC	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TN	170	170	170	170	170	170	170	170	170	170	170	170	170	170
TX	23194	23194	23194	23194	23194	23194	23194	23194	23194	23194	23194	23194	23194	23194
UT	166	166	166	166	166	166	166	166	166	166	166	166	166	166
VA	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WA	0	0	0	0	0	0	0	0	0	0	0	0	0	0

In [72]:

```
# Compute summary statistics of time differences
stats_post = remits_pre_covid.groupby('Billing Provider
State')['num_days'].agg(['mean', 'min', 'max', 'std'])
```

```
stats_post.compute()
```

Out[72]:

	mean	min	max	std
Billing Provider State				
AZ	NaN	NaN	NaN	NaN
CA	0.256000	0.0	9.0	1.021095
CO	NaN	NaN	NaN	NaN
FL	0.579412	0.0	407.0	7.318902
GA	0.250000	0.0	1.0	0.500000
IA	NaN	NaN	NaN	NaN
ID	NaN	NaN	NaN	NaN
IL	NaN	NaN	NaN	NaN
IN	0.503423	0.0	182.0	2.129228
LA	0.555556	0.0	1.0	0.527046
MA	NaN	NaN	NaN	NaN
ME	NaN	NaN	NaN	NaN
MI	0.769343	0.0	20.0	2.293178
MO	0.875776	0.0	16.0	2.388336
NC	0.557480	0.0	19.0	1.557850
NE	NaN	NaN	NaN	NaN
NH	0.449735	0.0	6.0	0.990063
NV	0.000000	0.0	0.0	0.000000
NY	0.000000	0.0	0.0	0.000000
OA	0.000000	0.0	0.0	0.000000
OH	NaN	NaN	NaN	NaN

	mean	min	max	std
Billing Provider State				
OK	NaN	NaN	NaN	NaN
PA	1.451109	0.0	8172.0	86.907565
Pr	NaN	NaN	NaN	NaN
SC	NaN	NaN	NaN	NaN
TN	0.364706	0.0	9.0	1.262542
TX	1.169225	0.0	7277.0	67.302929
UT	0.722892	0.0	6.0	1.272680
VA	NaN	NaN	NaN	NaN
WA	NaN	NaN	NaN	NaN

INTERPRETATION: The Pennsylvania state took more time to complete the payement throughout all status. New Hampshire, Nevada, New York took least time for completing the payment

Post-covid Era

Chi-Squared Test between Claim Status Category and Billing Provider State

Hypothesis

Null Hypothesis: There is no relationship between Claim Status Category and Billing Provider State

Alternate Hypothesis: There is a relationship between Claim Status Category and Billing Provider State

In [77]:

```
# perform a chi-squared test on two categorical variables
contingency_table = pd.crosstab(remits_post_covid['Billing Provider State'],
remits_post_covid['Claim Status Category'])
chi2, p, dof, expected = ss.chi2_contingency(contingency_table)

print(f"Chi-squared: {chi2:.2f}")
print(f"p-value: {p:.2f}")
print(f"Degrees of freedom: {dof}")
Chi-squared: 12833.94
p-value: 0.00
Degrees of freedom: 84
```

INTERPRETATION: Since the P-value is less than 0.05, we reject null hypothesis. So, there is a relationship between billing provider state and claim status category.

In [78]:

```
# calculate the standardized residuals
standardized_residuals = (contingency_table - expected) / expected.std()

print(standardized_residuals)
col_0    Denied  Forwarded to Other Payer  Processed  Reversal
row_0
AZ    -0.000766             -7.805223e-06   0.000121  0.000653
CA     0.020599             -1.891970e-04  -0.027636  0.007226
CO    -0.000100             -9.557416e-07   0.000155 -0.000054
FL    -0.104876             -1.103722e-03   0.100283  0.005697
GA    -0.000058             -5.575159e-07   0.000090 -0.000031
IA    -0.000008             -7.964513e-08   0.000013 -0.000004
ID    -0.000004             -3.982257e-08   0.000006 -0.000002
IL    -0.000029             -2.787580e-07   0.000045 -0.000016
IN     0.015359             -8.345494e-03   0.096067 -0.103080
LA    -0.001965             -1.927412e-05   0.002076 -0.000092
MA    -0.000033             -3.185805e-07   0.000052 -0.000018
ME    -0.000008             -7.964513e-08   0.000013 -0.000004
MI    -0.022904             -2.191436e-04   0.023492 -0.000370
MO    -0.018900             -1.817900e-04   0.018292  0.000790
NC     0.048988             -8.482207e-05  -0.044227 -0.004676
NE    -0.000050             -4.778708e-07  -0.000072  0.000122
NH    -0.016806             -1.679318e-04   0.017788 -0.000814
NV     0.013951             -1.541920e-04  -0.022751  0.008953
OA     0.000045             -3.982257e-08  -0.000043 -0.000002
OH    -0.000008             -7.964513e-08   0.000013 -0.000004
OK    -0.000004             -3.982257e-08   0.000006 -0.000002
PA     0.086910              6.426865e-03  -0.089390 -0.003947
Pr    -0.000008             -7.964513e-08   0.000013 -0.000004
SC    -0.000012             -1.194677e-07   0.000019 -0.000007
TN    -0.006665             -8.705213e-05   0.006041  0.000711
TX    -0.010106              4.182860e-03  -0.083120  0.089043
UT    -0.002498             -4.575613e-05   0.002637 -0.000093
VA    -0.000033             -3.185805e-07   0.000052 -0.000018
WA    -0.000008             -7.964513e-08  -0.000037  0.000045
```

INTERPRETATION: Since standardized residual values for denied, processed, and reversal claims are between -2 and 2 indicating that all cells in the contingency table is not contributing significantly to the overall chi-square test statistic. The standardized residual values for all states except california, Florida, Lousiana, Missouri, New Hampshire, Nevada, South Carolina are less than -2, so are contributing significantly to overall chi-square test statistic.

In [79]:

```
# convert counts to proportions
prop_tab = contingency_table.div(contingency_table.sum(axis=1), axis=0)

# calculate odds ratios
ref_category = remits_post_covid['Billing Provider State']
odds_ratios = prop_tab.div(prop_tab.loc[ref_category])
```

```
# display the odds ratios
print(odds_ratios)
col_0  Denied  Forwarded to Other Payer  Processed  Reversal
row_0
AZ        1.0                       NaN        1.0       1.0
AZ        1.0                       NaN        1.0       1.0
AZ        1.0                       NaN        1.0       1.0
AZ        1.0                       NaN        1.0       1.0
AZ        1.0                       NaN        1.0       1.0
...       ...                       ...        ...       ...
VA        NaN                       NaN        1.0       NaN
VA        NaN                       NaN        1.0       NaN
VA        NaN                       NaN        1.0       NaN
WA        NaN                       NaN        1.0       1.0
WA        NaN                       NaN        1.0       1.0

[451417 rows x 4 columns]
```

INTERPRETATION: Since the odds ratios is 1 there is slightly a positive association between all the groups of claims status category and Billing Provider State

Assumptions of chi-squared: 1) All observations should be independent. 2) All the variable should be nominal or ordinal 3) The expected values should be 5 or higher in at least 80% of groups.

In [80]:

```
grouped_denial = remits_post_covid.groupby('Billing Provider State')

# Calculate the expected values for each group
expected_denial = grouped_denial.mean()

# Check the assumption that 80% of groups have expected values of 5 or higher
num_above_threshold = (expected_denial >= 5).sum().compute()
num_groups = len(expected_denial)
percent_above_threshold = num_above_threshold / num_groups * 100
percent_above_threshold
if (percent_above_threshold >= 80.0).any():
    print("Assumption met!")
else:
    print("Assumption not met.")
Assumption met!
```

INTERPRETATION:

- Since, all the observations are collected independently, the assumption all observations are independent is met.
- As the observations are measured as nominal, the assumption of all the variable should be nominal or ordinal is met.
- The assumption of The expected values should be 5 or higher in at least 80% of groups is not met

In [81]:

```
denied_claims_grouped = remits_post_covid.groupby(['Billing Provider State'])
denied_claims_grouped.count().compute()
```

Out[81]:

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
AZ	196	196	196	196	196	196	196	196	196	196	196	196	196	196
CA	4751	4751	4751	4751	4751	4751	4751	4751	4751	4751	4751	4751	4751	4751
CO	24	24	24	24	24	24	24	24	24	24	24	24	24	24
FL	27716	27716	27716	27716	27716	27716	27716	27716	27716	27716	27716	27716	27716	27716
GA	14	14	14	14	14	14	14	14	14	14	14	14	14	14
IA	2	2	2	2	2	2	2	2	2	2	2	2	2	2
ID	1	1	1	1	1	1	1	1	1	1	1	1	1	1
IL	7	7	7	7	7	7	7	7	7	7	7	7	7	7
IN	215802	215802	215802	215802	215802	215802	215802	215802	215802	215802	215802	215802	215802	215802
LA	484	484	484	484	484	484	484	484	484	484	484	484	484	484
MA	8	8	8	8	8	8	8	8	8	8	8	8	8	8
ME	2	2	2	2	2	2	2	2	2	2	2	2	2	2
MI	5503	5503	5503	5503	5503	5503	5503	5503	5503	5503	5503	5503	5503	5503

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
MO	4565	4565	4565	4565	4565	4565	4565	4565	4565	4565	4565	4565	4565	4565
NC	2130	2130	2130	2130	2130	2130	2130	2130	2130	2130	2130	2130	2130	2130
NE	12	12	12	12	12	12	12	12	12	12	12	12	12	12
NH	4217	4217	4217	4217	4217	4217	4217	4217	4217	4217	4217	4217	4217	4217
NV	7613	7613	7613	7613	7613	7613	7613	7613	7613	7613	7613	7613	7613	7613
NY	0	0	0	0	0	0	0	0	0	0	0	0	0	0
OA	1	1	1	1	1	1	1	1	1	1	1	1	1	1
OH	2	2	2	2	2	2	2	2	2	2	2	2	2	2
OK	1	1	1	1	1	1	1	1	1	1	1	1	1	1
PA	66815	66815	66815	66815	66815	66815	66815	66815	66815	66815	66815	66815	66815	66815
Pr	2	2	2	2	2	2	2	2	2	2	2	2	2	2
SC	3	3	3	3	3	3	3	3	3	3	3	3	3	3
TN	2186	2186	2186	2186	2186	2186	2186	2186	2186	2186	2186	2186	2186	2186

	Statement End FY	Statement Start Date	Statement End Date	RMA_Payer: Prim Lvl III	Denial Type	Claim Status Description	Claim Status Category	Claim Received Date	Claim Filling Code	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
Billing Provider State														
TX	108201	108201	108201	108201	108201	108201	108201	108201	108201	108201	108201	108201	108201	108201
UT	1149	1149	1149	1149	1149	1149	1149	1149	1149	1149	1149	1149	1149	1149
VA	8	8	8	8	8	8	8	8	8	8	8	8	8	8
WA	2	2	2	2	2	2	2	2	2	2	2	2	2	2

In [83]:

```
# Compute summary statistics of time differences
stats_post = remits_post_covid.groupby('Billing Provider
State')['num_days'].agg(['mean', 'min', 'max', 'std'])
stats_post.compute()
```

Out[83]:

	mean	min	max	std
Billing Provider State				
AZ	0.403061	0.0	6.0	0.942333
CA	0.190065	0.0	183.0	2.756709
CO	0.000000	0.0	0.0	0.000000
FL	0.211069	0.0	152.0	2.076919
GA	0.642857	0.0	7.0	1.864946
IA	0.000000	0.0	0.0	0.000000

	mean	min	max	std
Billing Provider State				
ID	0.000000	0.0	0.0	NaN
IL	0.000000	0.0	0.0	0.000000
IN	0.203279	0.0	291.0	1.190796
LA	0.055785	0.0	2.0	0.263335
MA	0.250000	0.0	2.0	0.707107
ME	0.000000	0.0	0.0	0.000000
MI	0.260767	0.0	26.0	1.065025
MO	0.118510	0.0	16.0	0.671977
NC	0.918779	0.0	25.0	2.733243
NE	0.000000	0.0	0.0	0.000000
NH	0.193502	0.0	157.0	2.531495
NV	0.263628	0.0	65.0	1.495303
NY	NaN	NaN	NaN	NaN
OA	0.000000	0.0	0.0	NaN
OH	0.000000	0.0	0.0	0.000000
OK	0.000000	0.0	0.0	NaN
PA	0.262621	0.0	232.0	1.532443
Pr	0.000000	0.0	0.0	0.000000
SC	0.000000	0.0	0.0	0.000000
TN	0.172004	0.0	13.0	0.779992
TX	0.196717	0.0	28.0	0.999445
UT	0.181027	0.0	9.0	0.702377
VA	0.000000	0.0	0.0	0.000000

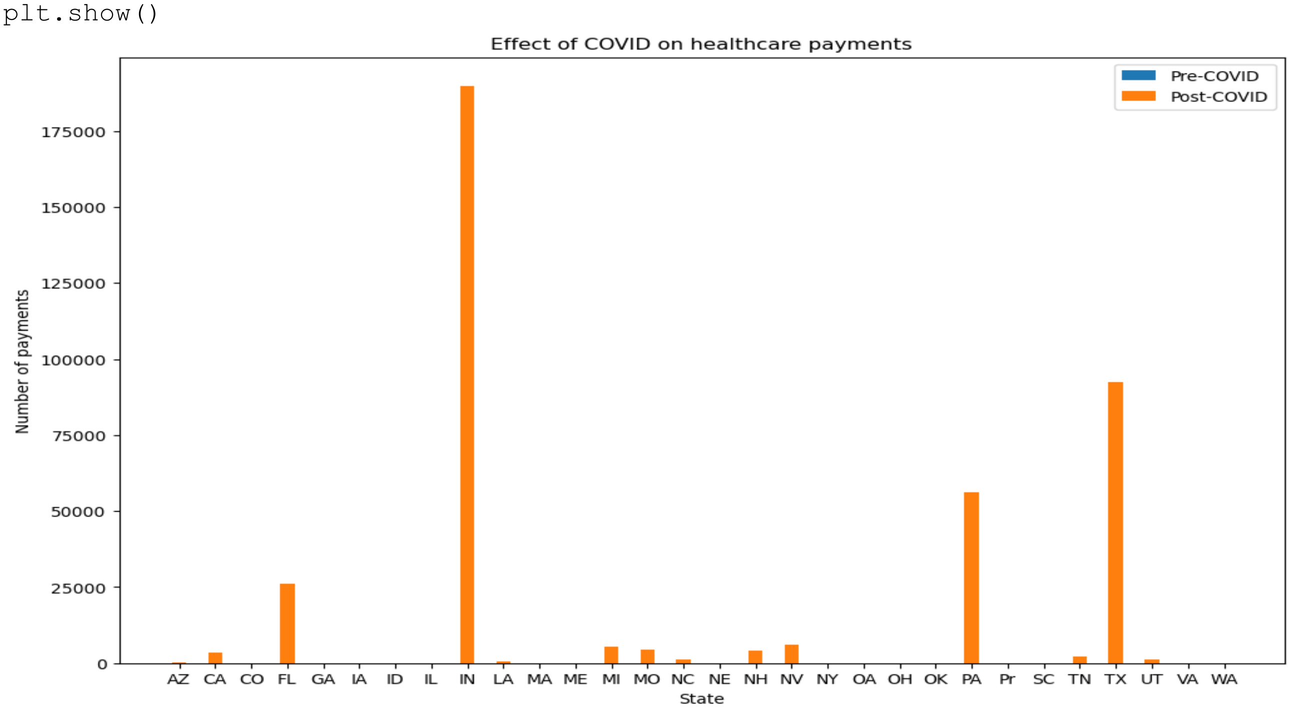
	mean	min	max	std
Billing Provider State				
WA	1.000000	1.0	1.0	0.000000

INTERPRETATION: Indiana took more time to make the payment, Colorado, Iowa, Idaho, Illinois, Maine, Nebraska, Ohio, Oklahoma,South Carolina, Virginia, Peuto Rico took very less time.

Visualizing the effect of covid on payments by state

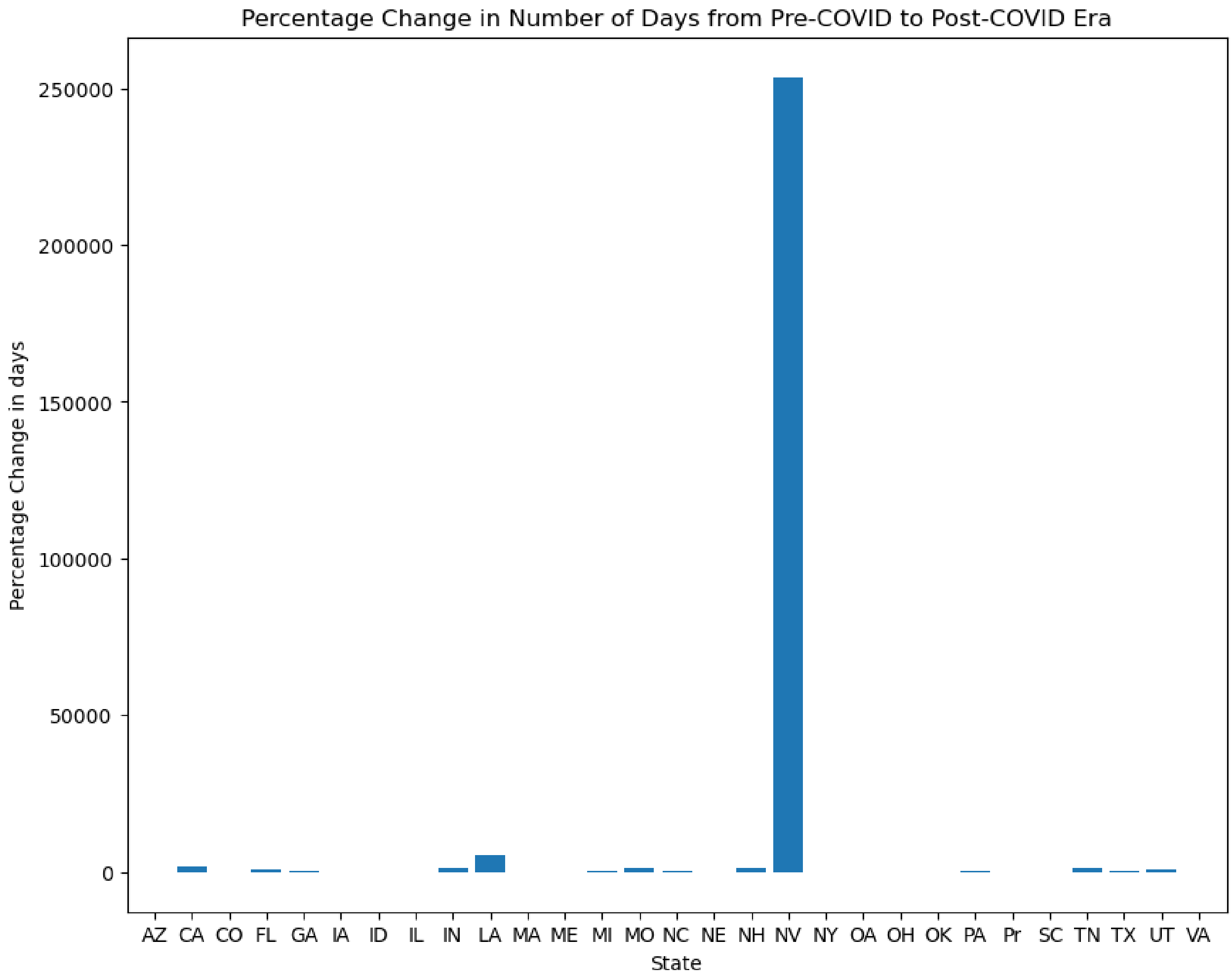
In [90]:

```
# Group the data by state and payment status (paid or not paid)
pre_covid_payments_grouped = remits_pre_covid.groupby(['Billing Provider
State', 'Claim Status Category']).size().reset_index().compute()
post_covid_payments_grouped = remits_post_covid.groupby(['Billing Provider
State', 'Claim Status Category']).size().reset_index().compute()

# Plot the payment status (paid or not paid) against the state to visualize
the effect of COVID on healthcare payments
fig, ax = plt.subplots(figsize=(12, 8))
ax.bar(pre_covid_payments_grouped['Billing Provider State'],
pre_covid_payments_grouped[0], width=0.4, label='Pre-COVID')
ax.bar(post_covid_payments_grouped['Billing Provider State'],
post_covid_payments_grouped[0], width=0.4, label='Post-COVID')
ax.set_xlabel('State')
ax.set_ylabel('Number of payments')
ax.set_title('Effect of COVID on healthcare payments')
ax.legend()
plt.show()
```



In [113]:

```
# Group by state and calculate total healthcare expenditure for pre-COVID era
remits_pre_covid_total = remits_pre_covid.groupby('Billing Provider
State')['num_days'].count().reset_index().compute()

# Group by state and calculate total healthcare expenditure for post-COVID
era
remits_post_covid_total = remits_post_covid.groupby('Billing Provider
State')['num_days'].count().reset_index().compute()

# Merge the dataframes
remits_combined = ddf.merge(remits_pre_covid_total, remits_post_covid_total,
on='Billing Provider State', suffixes=('_pre', '_post'))

# Calculate percentage change in number of days
remits_combined['percentage_change'] = ((remits_combined['num_days_post'] -
remits_combined['num_days_pre']) / remits_combined['num_days_pre']) * 100

# Visualize the data
plt.figure(figsize=(10, 8))
plt.bar(remits_combined['Billing Provider State'],
remits_combined['percentage_change'])
plt.xlabel('State')
plt.ylabel('Percentage Change in days')
plt.title('Percentage Change in Number of Days from Pre-COVID to Post-COVID
Era')
plt.show()
```



INTERPRETATION: The percentage change in number of days taken to complete the payment is larger in Nevada, and there is no change in South Carolina, Puerto Rico, Oklahoma, Ohio, New York, Nebraska, Manie, Massachusetts, Illinois, Idaho, Iowa, Colorado, and Arizona.

Report: There is a relationship between the Billing Provider State and the Claim Status Category,and the payment time is improved post covid.

PREDICTIVE ANALYSIS

Research question 4: Predicting if the claims are paid or not

To predict if the claims are paid or not, the predictor variables and the outcome variable Claim Status Category are divided and the machine learning models Logistic Regression, Guassian Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost, Xg Boost are fitted. The accuracy and time taken to fit model is calculated.

In [92]:

```
## computing the data frame

df = remits.compute()
```

In [93]:

```
## dividing the dataset

X,Y=df[['Statement End FY','RMA_Payer: Prim Lvl III','Claim Status
Description','Denial Type',
        'Claim Filling Code','Billing Provider State','Billing Provider
City','Initial Denial Remit Count',
        'Total Denied Remit Count','Final Denials Remit
Count','num_days']],df['Claim Status Category']
```

In [94]:

```
## classifying the outcome variable as denied or not denied.

Y=[1 if y == 'Denied' else 0 for y in Y]
```

In [95]:

```
## applying encoding to the predictor variables

X=X.apply(LabelEncoder().fit_transform)
```

In [96]:

```
X
```

Out[96]:

	Statement End FY	RMA_Payer: Prim Lvl III	Claim Status Description	Denial Type	Claim Filling Code	Billing Provider State	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
0	0	0	2	15	1	26	15	0	0	0	0
1	0	0	2	15	2	3	22	0	0	0	0
2	0	0	7	15	1	26	15	0	0	0	0
3	0	0	7	15	2	3	22	0	0	0	0
4	0	0	2	17	8	8	34	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...
586139	4	3	2	15	2	8	34	0	0	0	0

	Statement End FY	RMA_Payer: Prim Lvl III	Claim Status Description	Denial Type	Claim Filling Code	Billing Provider State	Billing Provider City	Initial Denial Remit Count	Total Denied Remit Count	Final Denials Remit Count	num_days
586140	4	3	2	15	12	8	34	0	0	0	0
586141	4	3	3	15	11	8	34	0	0	0	0
586146	4	3	2	2	4	22	78	1	1	0	43
586148	4	0	2	2	4	22	78	1	1	0	49

516959 rows × 11 columns

In [97]:

```
## splitting the data set into 80% training and 20% testing dataset

X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.2)
```

In [98]:

```
## fitting the logistic regression

start_time = time.time()

model=LogisticRegression()
model.fit(X_train,Y_train)

## predicting on testing data set

ypred=model.predict(X_test)
```

```
end_time = time.time()
print("Time taken: %0.2f seconds" % (end_time - start_time))

## calculating accuracy

print(accuracy_score(Y_test,ypred))
Time taken: 3.42 seconds
1.0
```

In [99]:

```
start_time = time.time()

## fitting the Naive Bayes

model=GaussianNB()
model.fit(X_train,Y_train)

## predicting on testing data set

ypred=model.predict(X_test)

end_time = time.time()
print("Time taken: %0.2f seconds" % (end_time - start_time))

## calculating accuracy

print(accuracy_score(Y_test,ypred))
Time taken: 0.46 seconds
1.0
```

In [100]:

```
## fitting the K Nearest Neighbors

start_time = time.time()

model=KNeighborsClassifier()
model.fit(X_train,Y_train)

## predicting on testing data set

ypred=model.predict(X_test)

end_time = time.time()
print("Time taken: %0.2f seconds" % (end_time - start_time))

## calculating accuracy

print(accuracy_score(Y_test,ypred))
```

```
Time taken: 87.31 seconds
0.9988877282575054
```

In [101]:

```
## fitting the Support Vector Machines

start_time = time.time()

model=SVC()
model.fit(X_train,Y_train)

## predicting on testing data set
```

```
ypred=model.predict(X_test)


end_time = time.time()

print("Time taken: %0.2f seconds" % (end_time - start_time))


## calculating accuracy


print(accuracy_score(Y_test,ypred))

Time taken: 960.21 seconds

0.9999903280718044
```

In [102]:

```
## fitting the Random Forest


start_time = time.time()


model=RandomForestClassifier()

model.fit(X_train,Y_train)


## predicting on testing data set


ypred=model.predict(X_test)


end_time = time.time()

print("Time taken: %0.2f seconds" % (end_time - start_time))


## calculating accuracy


print(accuracy_score(Y_test,ypred))

Time taken: 17.33 seconds
```

```
1.0
```

In [103]:

```
## fitting the Gradient Boosting

start_time = time.time()

model=GradientBoostingClassifier()
model.fit(X_train,Y_train)

## predicting on testing data set

ypred=model.predict(X_test)

end_time = time.time()
print("Time taken: %0.2f seconds" % (end_time - start_time))

## calculating accuracy

print(accuracy_score(Y_test,ypred))
```

```
Time taken: 21.85 seconds
1.0
```

In [104]:

```
## fitting the XG Boost

start_time = time.time()

model=XGBClassifier()
model.fit(X_train,Y_train)
```

```
## predicting on testing data set

ypred=model.predict(X_test)

end_time = time.time()
print("Time taken: %0.2f seconds" % (end_time - start_time))

## calculating accuracy

print(accuracy_score(Y_test,ypred))
Time taken: 5.74 seconds
1.0
```

INTEPRETATION: The time taken for fitting the model and the accuracy of logistic regression are 3.42 seconds, and 100 percent respectively. The time taken for fitting the model and the accuracy of Gaussian Naive Bayes are 0.46 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of K-Nearest Neighbors are 87.31 seconds and 99.8 percent respectively. The time taken for fitting the model and the accuracy of Support Vector Machines are 960.21 seconds 99.9 percent respectively. The time taken for fitting the model and the accuracy of Random Forest are 17.33 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of Gradient Boost are 21.85 seconds and 100 percent respectively. The time taken for fitting the model and the accuracy of XG Boost are 5.74 seconds and 100 percent respectively. Thus, the best model is Gaussian Naive Bayes as it took less time and provided 100 percent accuracy. The other better models are logistic regression, XG boost, Random Forest, Gradient Boost respectively.

Research question 5: Predicting time taken to make the entire payment

To make the predictions on time taken to make the payment a linear regression model is fit and score is calculated. This model is used on the claims data set to calculate the time.

In [118]:

```
df=remits.compute()

# Filter the insurance claim records
accepted_insurance_claims = df[df['Claim Status Category'] != 'Denied' ]
```

```
X,Y=accepted_insurance_claims[['RMA_Payer: Prim Lvl III','Billing Provider
State','Billing Provider City',]],

    accepted_insurance_claims[['num_days']]

X=X.apply(LabelEncoder().fit_transform)


# Split the data into training and testing sets


X_train, X_test, Y_train, Y_test=train_test_split(X,Y,test_size=0.2)


# convert dask DataFrame to Dask array


X_dask = da.from_array(X.values, chunks=(1000, X.shape[1]))

Y_dask = da.from_array(Y.values, chunks=1000)


# Fit a linear regression model

model = LinearRegression()

model.fit(X_dask, Y_dask)


# Evaluate the model on the testing data



X_test_processed = da.from_array(X.values, chunks=(1000, X.shape[1]))

Y_test_processed = da.from_array(Y.values, chunks=1000)


y_prediction = model.predict(X_test_processed)

score = model.score(X_test_processed, Y_test_processed)
```

In [116]:

```
## Making  predictions on claims data
```

```
claims_insurance = claims[['RMA_Payer: Prim Lvl III','Billing Provider
State','Billing Provider City']].compute()


X_new = claims_insurance


X= claims_insurance[['RMA_Payer: Prim Lvl III','Billing Provider
State','Billing Provider City']]

X=X.apply(LabelEncoder().fit_transform)


X_dask = da.from_array(X.values, chunks=(1000, X.shape[1]))


y_pred = model.predict(X_dask)
```

CONCLUSION

The study used the Dask framework in Python and claims and denial data from the Envision Healthcare database to analyze trends in pre-COVID and post-COVID era. The datasets were cleaned by removing nulls and changing data types to appropriate types. Descriptive statistics were calculated for numerical attributes in both datasets, and visualizations like scatter plots, heatmaps, stacked bar charts, and bar charts were used to reveal interactions between variables. Statistical tests like Chi-Squared test and Multinomial regression were used to analyze the datasets, and machine learning classification models like Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Support Vector Machines, Random Forest, Gradient Boost, XG Boost were used to predict if claims were paid or not. A linear regression model was also used to predict the time taken for the payment. The study found a relationship between the type of insurance and claim status category, and that there was a relationship between the type of insurance and the number of days taken for payment. The study also found a relationship between claim status category and billing provider state, with a percentage change in the number of days from pre-COVID to post-COVID. In the predictive analysis it is found that the best model for predicting payment of claims was Random Forest, and the best model for predicting the time taken for payment was Linear Regression.

In [ ]: