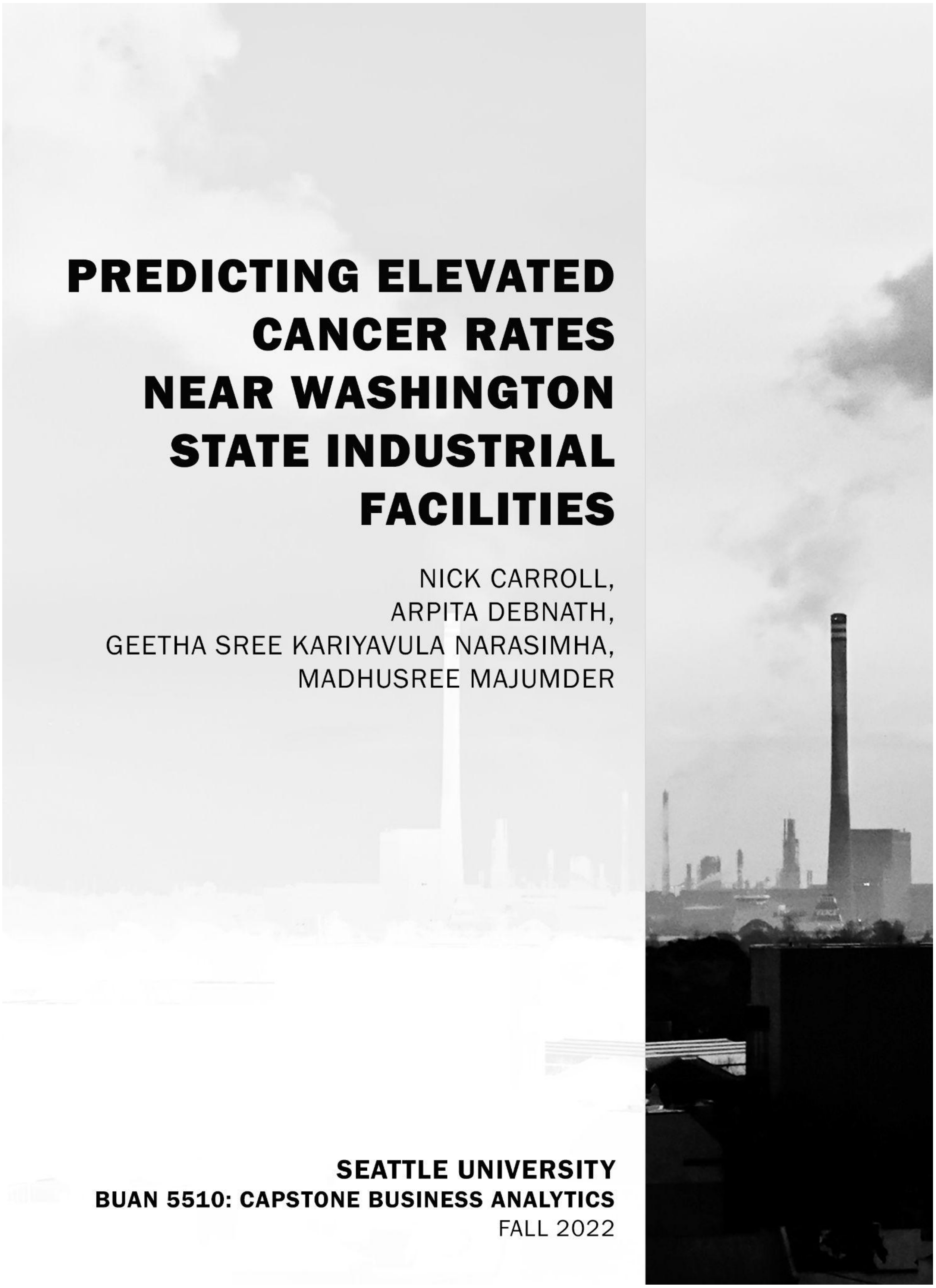


PREDICTING ELEVATED CANCER RATES NEAR WASHINGTON STATE INDUSTRIAL FACILITIES

NICK CARROLL,
ARPITA DEBNATH,
GEETHA SREE KARIYAVULA NARASIMHA,
MADHUSREE MAJUMDER

SEATTLE UNIVERSITY
BUAN 5510: CAPSTONE BUSINESS ANALYTICS
FALL 2022

Table of Contents

Abstract

Health is of utmost importance to every living being. Toxic releases from factories cause serious illness which could have mild impact or could even be fatal. For the current and upcoming generations to lead a healthy life, it is necessary to have the impact of toxic releases on us to be as minimal as possible.
United States Environmental Protection Agency has provided data of Washington state for 2021 year with details regarding Facility Name, Industry Sector, Chemicals released etc which can be used for detailed analysis of understanding how the toxic releases are impacting health by the presence of carcinogens at a facility site. A carcinogen is a substance, organism or agent capable of causing cancer.

Machine learning algorithms have played a significant role in the Data Science field in providing both predictive and descriptive analysis for many problems. Similarly, we have used the current data using machine learning algorithms to predict the impact of toxic releases on health as indicated by the presence of carcinogens at a facility site. We started with the TRI report 2021 and performed all data pre processing. Later we added a new dataset from the Cancer Registry from Washington State Department of Health, which tracks incidents of cancer across counties as well as that county's estimated annual population.

With this new dataset post merging first we have proceeded with hyperparameter tuning and then built five machine learning models- Decision Tree, Random Forest, Gradient Boosting, Naive Bayesian Classification and Neural Network model. Random forest classifier performed the best among all, especially for SMOTE applied dataset with 97% accuracy and 98% Recall.

Introduction

Problem Statement

Predicting if a county has an elevated cancer rate as indicated by data provided by Toxic Release Inventory on Washington State Industrial Facilities.

Contributions

Since 1990, global cancer prevalence has risen from 54% to 64% (Our World in Data). Exposure to certain chemicals in the environment, at home, and at work may contribute to an individual's risk of developing cancer. Initially we started with the dataset obtained from TRI to predict the presence of carcinogen in the facility's releases. But then, we extended our research to evaluate if a county has an elevated cancer rate as indicated by data provided by Toxic Release Inventory on Washington State Industrial Facilities. For this, we merged the dataset obtained from TRI with cancer registry from the Washington State Department of Health for the next phase. TRI dataset is mostly about details regarding releases from the facilities which includes chemicals that are present, whether the chemical is carcinogenic or not, whether the release contains metal components or not, location of the facility, etc. While the dataset obtained from the cancer registry has details on annual population, annual observations, etc. These two datasets have been joined by year.

There has been substantial Research conducted linking industrial factories to increased rates of many different forms of cancers. Thus, we are evaluating if a county has an elevated cancer rate as indicated by data provided by Toxic Release Inventory on Washington State Industrial Facilities.

Background Information

The Toxic Release Inventory (TRI) Program are annual reports administered by the United States Environmental Protection Agency that track facilities that release toxic-chemicals which may pose a threat to the environment and human health. Over 700 chemicals are individually tracked.

We have also added a new dataset from the Cancer Registry from Washington State Department of Health, which tracks incidents of cancer across counties as well as that county's estimated annual population, from 1994 to 2018.

The Datasets

Our datasets contained nominal and numerical attributes. In this section, we'll provide a description and exploration of our two datasets, which are Toxic Release Inventory Reports and Washington State Cancer Registry data. For each of the two datasets, we'll discuss and provide a broader overview of the datasets and discuss a selection of attributes that provide a broader overview of the datasets. Additional analysis on attributes not covered is included in the Appendix.

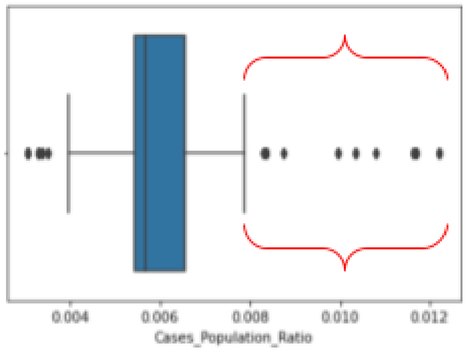
Toxics Release Inventory

The Toxics Release Inventory (TRI) is a dataset compiled by the U.S. Environmental Protection Agency (EPA). TRI reporting is an annual report done by businesses that manufacture, process, or otherwise use hazardous chemicals as part of their operations. It contains information on the release and waste management for over 800 toxic chemicals and toxic chemical categories as reported annually by facilities in certain industries as well as federal facilities.

Dataset Description

In this project, we used The Toxic Release Inventory (TRI) Reporting Form R and Form A Certification Statement Data for Washington State Facilities from 1995 to 2018 was acquired from the United States Environmental Protection Agency (EPA). This dataset consisted of annual tables in this period containing 26.5 MB of data across 124 attributes and 25,799 rows.

Figure 1: Cases / Population Ratio Boxplot



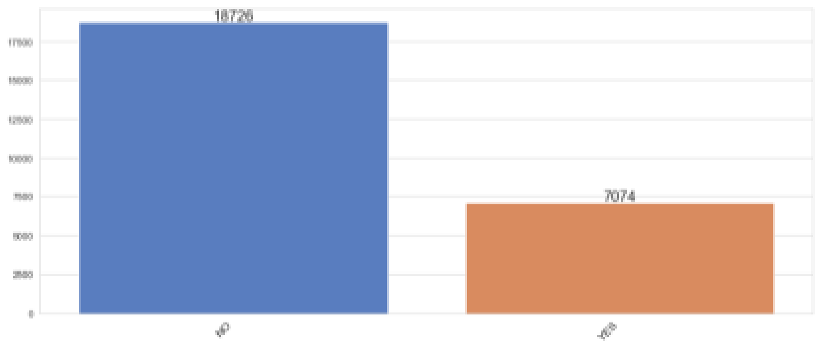
Note: Numbers in this range are Upper Range Extreme Outliers. These numbers form the basis of the 'Cancer Rate Flag' (see Target Variable Section).

Data Exploration

Numerical Data

We have created summary Statistics for all the numerical variables and for a few variables we observed very high standard deviation and skewness. So these data were transformed further by removing outliers. We have created the boxplot for these variables in order to see the skewness and outliers. Inorder to explore each numerical variable, we created the histograms to observe the distribution of the dataset.
There are 75 numerical variables out of which 40 variables have been discarded. Few of these variables have absolutely no data,
few of the variables have greater than 50% of missing data
while the others are not relevant to be part of the dataset (like Year, ID, latitude, longitude etc.)

Figure 2: Counts of TRI Reports where Carcinogens Reported

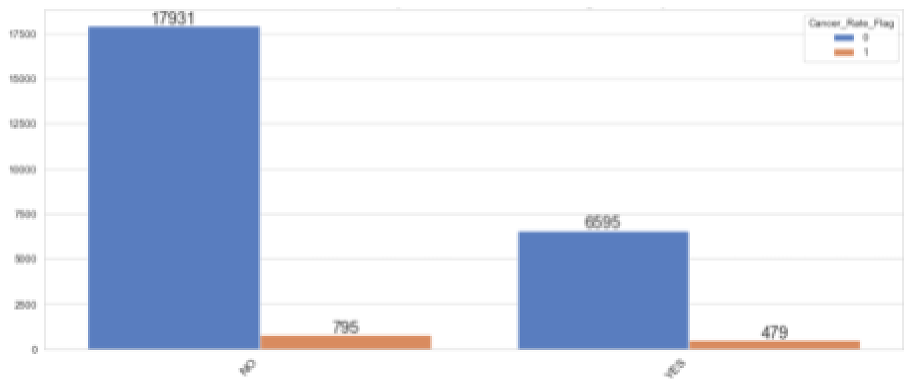


Nominal Data

Carcinogens

The initial data set obtained from the TRI has details on the chemicals present in releases from factories such as Ammonia, Manganese, Chromium, Naphthalene, n-Butyl alcohol etc. These releases from the facilities produce Carcinogens. The visualization in Figure 2 shows the counts of TRI Reports where Carcinogens were reported from facilities.

Figure 2.1: Count of TRI Reports Where Carcinogens Reported (and Cancer Rate Flag)

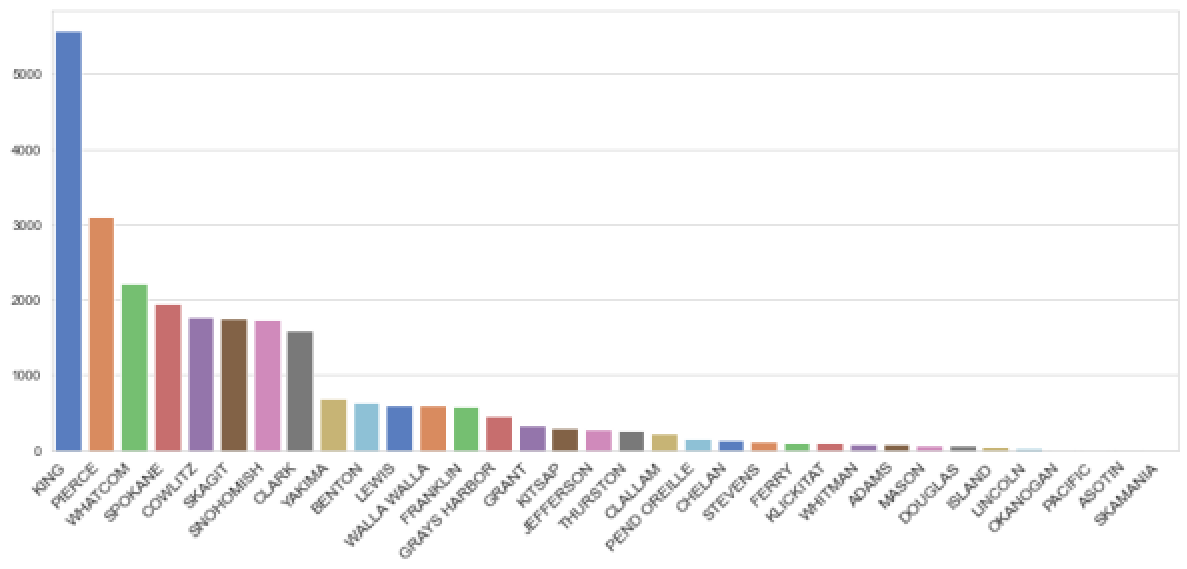


18,276 TRI reports did not mention Carcinogens, whereas 7,074 reports mentioned Carcinogens. When we began our research, this was the initial target variable. In Figure 2.1, we can see Carcinogens broken down further with the addition of the Cancer Rate Flag, which we will discuss later. Both the Cancer Rate Flag and Carcinogens both have an asymmetrical distribution.

County

TRI reports were analyzed from 34 out of the 39 Washington State counties. King & Pierce County had the first and second most TRI reports, in addition to being the two most populous counties in Washington State. However, Whatcom, which has the third largest number of TRI reports, is the 9th most populous county. Another county with a great difference between the number of TRI reports and the county's population is Cowlitz, which while it has the 5th highest number of TRI reports, has the 12th highest population. These numbers were based on 2018 population figures.

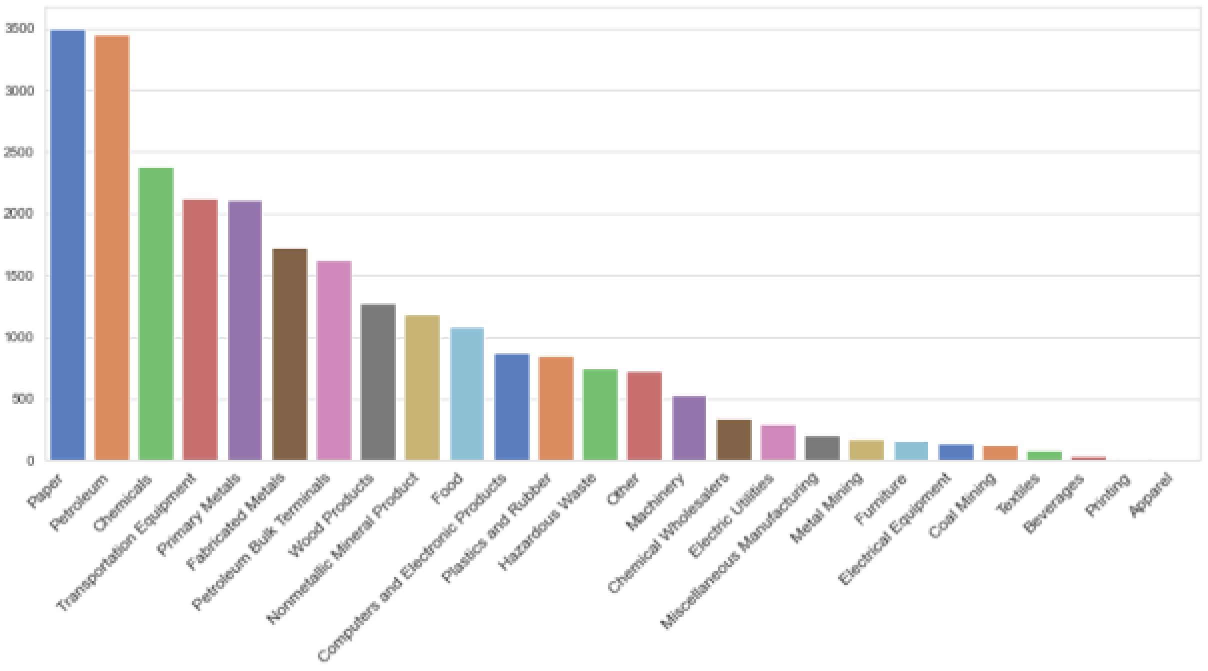
Figure 3: TRI Report Counts by County



Industry

There are 26 unique industries in this dataset. The industry sector which produced the highest number of TRI reports was paper, closely followed by Petroleum.

Figure 4: TRI Report Counts by Industry Sector



Chemicals

There were 182 different chemicals in the dataset. Lead Compounds were found the most, followed by Lead, and Nitrate Compounds. As there is such an extensive number of chemicals in our dataset, we will only highlight the top 10 most frequently occurring chemicals (see Figure 5). In the appendix, there is an extended list of all chemicals found.

Washington State Cancer Registry

Our second dataset comes from the Washington State Cancer Registry (WSCR) from the Washington State Department of Health for the years from 1995 to 2018. WSCR regularly receives information on people newly diagnosed with cancer and new information about previously diagnosed cancer cases. WSCR provides data on cancer of all types combined and the 24 most frequently diagnosed cancers. Information is also provided on "Hodgkin lymphoma" and "Larynx" because previous years of the WSCR Annual Report included them among the 24 most frequently diagnosed cancers.

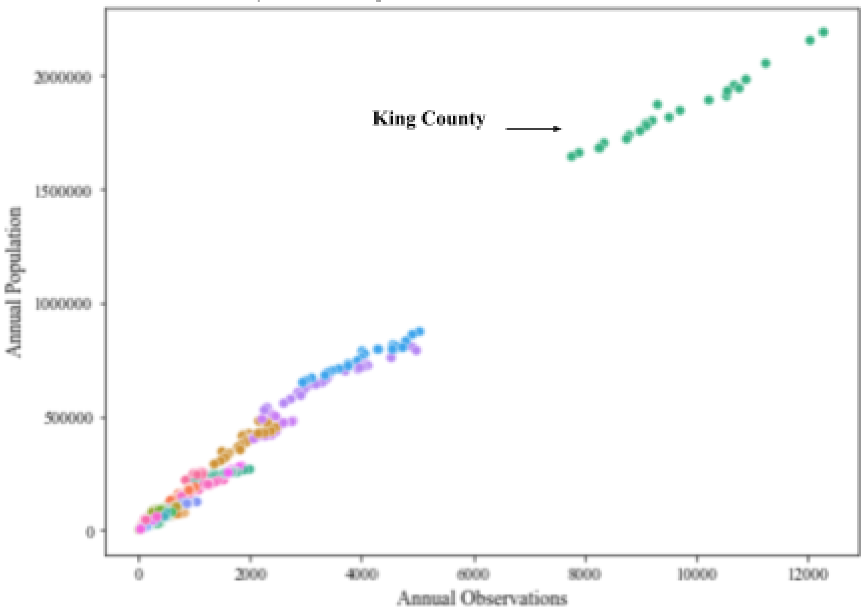
Dataset Description

This dataset consisted of annual tables of Washington State counties from 1995 to 2018 containing 109 KB of data 12 attributes and 897 rows.

Data Exploration

There are 23 Cancer Reports for each of the 39 counties in Washington state representing the 23 years between 1995 to 2018. Not surprisingly, a close relationship exists between annual population and annual cases for each county. The green dots represent King County, which by far has the highest number of annual cases and annual population regardless of year. The remainder of Washington State counties are clustered in the bottom left.

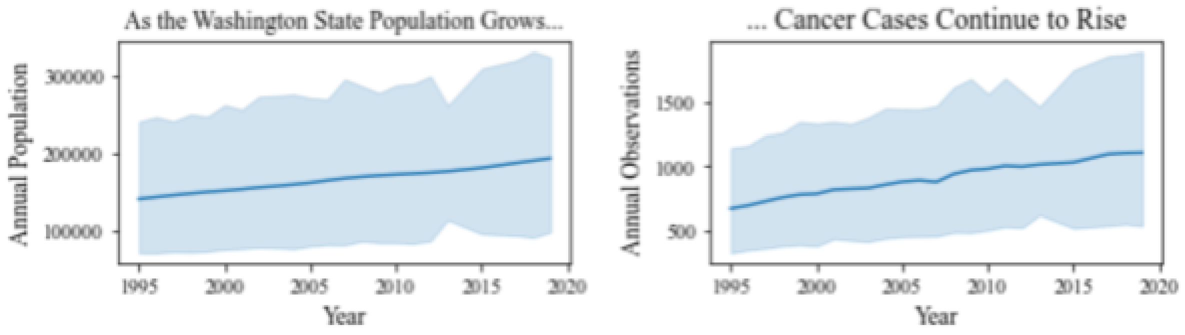
Figure 6: Cancer Cases Across Washington State Counties & Years
Note: Dots = Individual Year | Color = County



Also not surprisingly, as the population of Washington state grows, so do its cancer rates. In the following graphic, the dark blue line represents the mean across all counties, while the blue area shaded outside represents the range of values of population and cases among counties.

To create better cross-country comparisons, further feature engineering will be implemented and will be discussed later. The remainder of the attributes we won't discuss as they aren't relevant for this project or only contain either single or no values for the respective column.

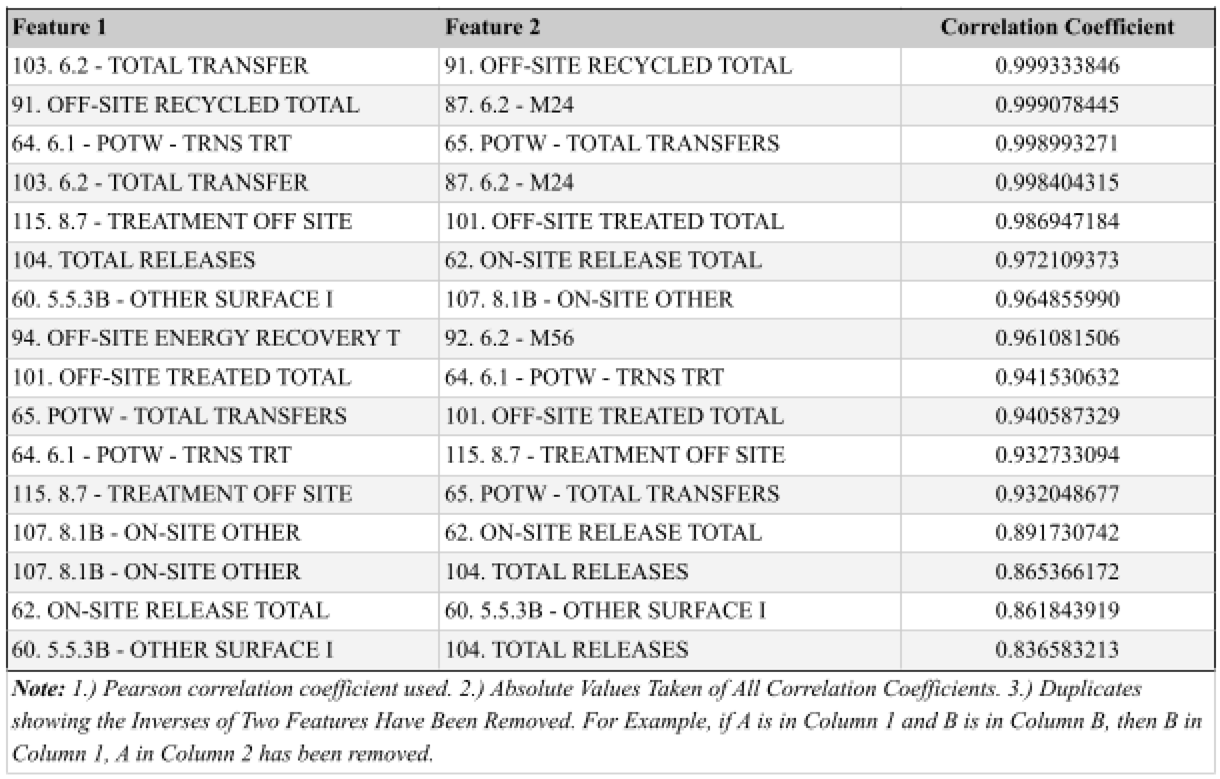
Figure 7: Growth of Cancer & Population in King County



Multivariate Analysis

The primary method utilized for conducting multivariate analysis in our dataset was through creating correlation matrices both assessing relationships among features, as well as through assessing correlations between solely the target variable and features. In Figure 8 we can see the correlation coefficients above 0.7. Due to the sheer size of our feature space, a full correlation matrix can be found in the appendix.

Figure 8: Correlation Coefficients of Numeric Data with Correlation Coefficient Above 0.7



Feature 1	Feature 2	Correlation Coefficient
103. 6.2 - TOTAL TRANSFER	91. OFF-SITE RECYCLED TOTAL	0.999333846
91. OFF-SITE RECYCLED TOTAL	87. 6.2 - M24	0.999078445
64. 6.1 - POTW - TRNS TRT	65. POTW - TOTAL TRANSFERS	0.998993271
103. 6.2 - TOTAL TRANSFER	87. 6.2 - M24	0.998404315
115. 8.7 - TREATMENT OFF SITE	101. OFF-SITE TREATED TOTAL	0.986947184
104. TOTAL RELEASES	62. ON-SITE RELEASE TOTAL	0.972109373
60. 5.5.3B - OTHER SURFACE I	107. 8.1B - ON-SITE OTHER	0.964855990
94. OFF-SITE ENERGY RECOVERY T	92. 6.2 - M56	0.961081506
101. OFF-SITE TREATED TOTAL	64. 6.1 - POTW - TRNS TRT	0.941530632
65. POTW - TOTAL TRANSFERS	101. OFF-SITE TREATED TOTAL	0.940587329
64. 6.1 - POTW - TRNS TRT	115. 8.7 - TREATMENT OFF SITE	0.932733094
115. 8.7 - TREATMENT OFF SITE	65. POTW - TOTAL TRANSFERS	0.932048677
107. 8.1B - ON-SITE OTHER	62. ON-SITE RELEASE TOTAL	0.891730742
107. 8.1B - ON-SITE OTHER	104. TOTAL RELEASES	0.865366172
62. ON-SITE RELEASE TOTAL	60. 5.5.3B - OTHER SURFACE I	0.861843919
60. 5.5.3B - OTHER SURFACE I	104. TOTAL RELEASES	0.836583213

Note: *1.) Pearson correlation coefficient used. 2.) Absolute Values Taken of All Correlation Coefficients. 3.) Duplicates showing the Inverses of Two Features Have Been Removed. For Example, if A is in Column 1 and B is in Column B, then B in Column 1, A in Column 2 has been removed.*

We identified many variables that were correlated with each other, but we expect that it will only be during the model tuning phase that we decide which of a set of two correlated features to include / not include.

<u>Literature Review</u>

1. **Toxic Exposure in America: Estimating Fetal and Infant Health Outcomes from 14 Years of TRI Reporting**

   (Nikhil Agarwal, Chanont Banternghansa, Linda T.M. Bui, 2010)

   This paper focuses on fetal and infant health outcomes due to toxic pollutants in the US from the year 1989 to 2002. Similar to our report, Toxic Release Inventory data was also included as an estimator. This study investigates health effects on two special groups: fetuses surviving at least 20 weeks in-utero and infants under one year of age. The author’s found significant health effects in infants, but not fetuses, due to toxic releases. Additionally, there is a statistically significant effect on infant mortality rates due to increase in toxic concentrations country wise. As such, the authors suggest the policy makers identify the toxic hazards and regulate their releases to the environment strictly to protect infants from adverse effects due to toxic release.

2. **Environmental Exposomics and Lung Cancer Risk Assessment in the Philadelphia Metropolitan Area Using ZIP Code–level Hazard Indices**

   (Thomas P. McKeon & Wei-Ting Hwang & Zhuoran Ding & Vicky Tam & Paul Wileyto & Karen Glanz & Trevor M. Penning, 2021)
   This paper focuses on lung cancer risk caused by environmental exposure and air pollutants in the Philadelphia metropolitan area, which has higher cancer rates than the national average. Although smoking is the main reason for lung cancer, other environmental exposure increases the risk of lung cancer among non-smokers and smokers as well. This paper used the modified multi-step multi-criteria decision analysis to investigate the causal effect. To derive results from the study, authors examined the emissions from Toxic Release Inventory, mapped cumulative exposures by feature, combined mean exposure by TRI features and NASA data and derived Hazard index from MMCDA. Two patterns were observed in this phase: some ZIP codes reported many chemicals with 100% fraction of occurrence, whereas several ZIP codes re-ported only one chemical but with 100% occurrence. The results showed that there were varying exposures across the 421 ZIP codes under study. ZIP codes with the highest hazard index tended to be in proximity to major highways which are important contributors to traffic-related air pollution in metropolitan areas.

3. **Toxicity Burden Score: A Novel Approach to Summarize Multiple Toxic Effects**

   (S.M.Lee1D.L.Hershman , P.Martin J.P.Leonard &Y.K.Cheung, March 2011)
   This paper is about Dose Limiting Toxicity (DLT) that means the side effects of a drug or other treatment that are serious enough to prevent an increase in dose or level of that treatment for Cancer drug trials. This paper has been written using the dataset from The National Cancer Institute—Common Terminology Criteria (NCI–CTC) for Adverse Events. This paper shows that TBS can be a feasible approach to summarize toxicity. To analyze that, regression approach has been used to obtain the severity weights and to summarize patient toxic effects into a TBS. The DLT has been redefined as TBS >=1 instead of setting a threshold for maximal toxicity It summarizes the information from the different grades and types of multiple toxic effects and can be helpful to apply in all phases of drug development.

4. **Toxic test scores: The impact of Chemical Releases on Standardized Test Performance Within U.S. Schools**

   (Irene Jacqz, September 2022)

In this paper the authors have described the effect of airborne chemicals toxic release on later-life cognitive performance from birth of a child. The effects of environmental pollution on student proficiency relies on local, temporal variation in airborne chemical releases during the years in which successive cohorts of students were born. This study shows how the composition of air pollution matters for long-term cognitive outcomes. It is also identified that early exposure to airborne particulates has adverse long-run effects on outcomes like test scores and earnings. In this case study- standardized test score data from the U.S. Department of Education's EDFacts(From 2009–2010 to 2014–2015) with the U.S. Environmental Protection Agency's Risk-Screening Environmental Indicators (RSEI) to form a six-year panel(from 2001 to 2006) dataset has been used.

5. **Total and Cardiovascular Mortality Rates in Relation to Discharges from Toxic Release Inventory Sites in the United States**

   (Hendryx, Luo, J., & Chen, B.-C, 2014)

   This study looks at the association between different variables covered in the Toxic Release Inventory and the mortality rates among both men and women. To conduct this study, the authors reviewed the TRI reports and averaged the county-level reported releases of toxic chemicals from these facilities and then 2-3 decades later looked at age-adjusted mortality rates in counties where these facilities are located. Four different categories of chemicals were studied: carcinogens, metals, hazardous air pollutants, and chemicals that are included in the Comprehensive Environmental Response, Compensation and Liability Act. Multiple linear regression models were implemented to assess the impact of these chemicals on mortality rates.

<u>Data Pre-Processing</u>

Merging Datasets

For each of the broader datasets from the Environmental Protection Agency and the Washington State Department of Health, each table contained one year of data. These years of data were appended on top of each other. From here, an inner join was performed on the two datasets on year and county.

Creating the Target Variable

Creating our target variable was a two part process. First we created a variable for Cancer Cases to Population Ratio. Second, a final target variable, "the Cancer Rate Flag", was created.

From the Washington State Department of Health Dataset, for each year in each county, the number of cases was divided by the total population in the county. This number was called the Case/Population Ratio. In this distribution, values above 0.00740078 are upper range extreme outliers. This number is calculated by multiplying the Interquartile range of a distribution by 1.5

and then adding the third quartile value. In percentage terms, the upper range extreme outlier value translates to .74%, which translates to a .74% chance of receiving a cancer diagnosis. Counties and Years which were found to be above this threshold are found in the following table. Rows that represented Upper Range Extreme Outliers were classified as 1, whereas rows below this threshold were classified as 0. To avoid multicollinearity, year and county were then removed from the dataset. Additional numeric variables which originated from the Cancer Registry dataset such as Annual Population, Annual Observations, Age-Adj. Rate per 100,000 and 95% CI were removed as well.

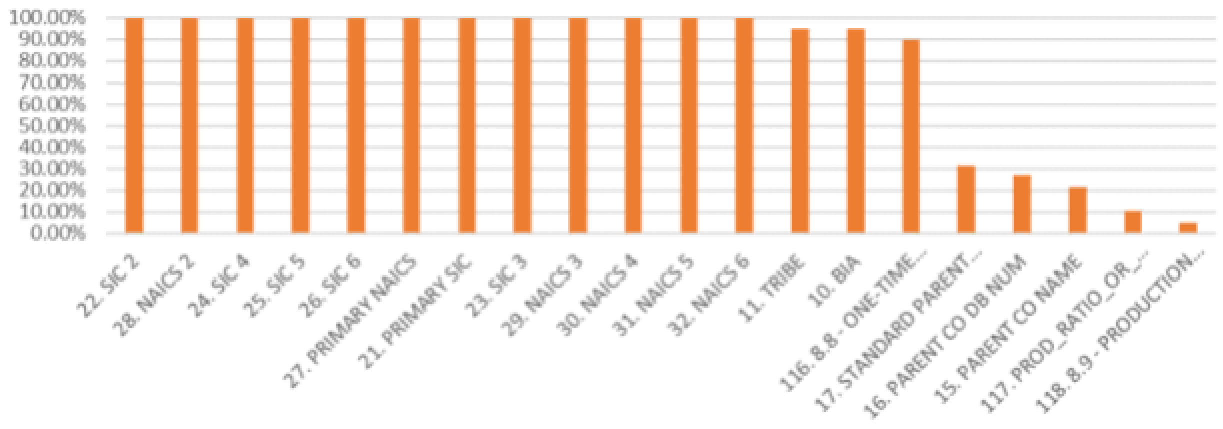
Figure 9: Years & Counties Classified as Upper Range Extreme Outliers

Year	Counties
1996	Jefferson, Mason
1997	Clallam
2000	Grays Harbor, Pend Oreille
2001	Lincoln, Pacific
2004	Ferry
2005	Island
2010	Okanogan
2011	Skagit
2013	Klickitat
2018	Kitsap, Chelan

Processing Null Values

There were multiple features that contained no values related to Standard Industrial Classification (SIC) Codes entered by facility, which had names such as SIC 2, 3, 4, 5, as well as NAICS, 3,4,5,6, which refers to the North American Industry Classification System. Two Features that had approximately 94% missing values were TRIBE and BIA, which relates to Tribal Identification. Typically a feature with this high of a percentage is removed, but upon inspecting some of the rows that didn't contain this value, it became clear that the absence of a value signaled that a facility wasn't on tribal land. As such, this was turned into a dummy variable for 'On Tribal Land' as 1 or "Not on Tribal Land' as zero. The Only Numeric Attribute with more than 50% missing values was One-Time Release, and there are other features related to this category specifically such as , TOTAL RELEASES., 8.1 - RELEASES, to name a few. After this in Figure 10, we can see 17. STANDARD PARENT CO NAME, 16. PARENT CO NAME and 'PARENT CO NAME' which are additional identifiers for facilities. The last numeric attribute with missing values is '118. 8.9 - PRODUCTION RATIO', with around 4% missing values.

Figure 10: Features with the Highest Percentage of Missing Values



Nominal Data Processing

Our dataset contained a wealth of data with regards to facility identifiers, chemical identifiers in TRI reports, and classifications for the types of waste and techniques utilized at facilities. Data which just provided additional irrelevant information identifying facilities such as city, state, and zip code were removed. This same logic was applied with additional chemical identifiers. Other nominal data was removed for different reasons as identified in the following table. The following table contains a detailed breakdown of the action taken for nominal attributes

Figure 11: Nominal Data Processing

Variables	Action	Additional Comments
YEAR,TRIFD, FRS ID, FACILITY NAME, STREET ADDRESS, CITY, COUNTY, ST, ZIP, BIA, HORIZONTAL DATUM, PARENT CO DB NUM, STANDARD PARENT CO NAME, FEDERAL FACILITY, INDUSTRY SECTOR CODE, INDUSTRY SECTOR, PARENT CO NAME	Discard	Additional facility identifiers and as such irrelevant attributes.
PRIMARY SIC, SIC 2, SIC 3, SIC 4, SIC 5, SIC 6, PRIMARY NAICS, NAICS 2, NAICS 3, NAICS 4, NAICS 5, NAICS 6, PFAS, DATA TYPE, CANCER SITE, STAGE AT DIAGNOSIS, AGE GROUP, GENDER, RACE,	Discard	Contained either zero or a single value.
TRI CHEMICAL/COMPOUND ID, SRS ID, METAL CATEGORY	Discard	Additional unnecessary unique ID for the chemical.
FORM TYPE	Discard	There are different forms used, which affects other columns. 56 Rows are dropped for other Form Type too.
UNIT OF MEASURE	Discard	22 Rows are in grams. The rest are in pounds. These 22 rows

		have been dropped too.
TRIBE, CLEAN AIR ACT CHEMICAL, CLASSIFICATION, METAL, CARCINOGEN, ELEMENTAL METAL INCLUDED, PROD_RATIO_OR_ACTIVITY, CAS#	One-Hot Encoding	
CHEMICAL	One-Hot Encoding	

As our dataset contains nominal but not ordinal data, one hot encoding was implemented for nominal attributes. There are two ways to convert categorical (nominal / ordinal) data to numerical data: Integer encoding and One-Hot Encoding.

Integer encoding works well for ordinal data where there is a set order in the data. Hence integer encoding does not work well for nominal data. One hot encoding works well for nominal data where the data is converted to a representation filled with 0s and 1s.

<u>One Hot Encoding For Nominal Variables</u>

One hot encoding is the process of converting nominal data variables to be provided to machine learning algorithms, which in turn improve predictions as well as classification accuracy of a model. In our dataset, we have several nominal variables. Using One Hot encoding, we have created dummy variables for these attributes. In general, one hot encoding provides better resolution of the data for the model and most models end up performing better.

Numeric Data Processing

In our dataset, we deleted numeric features containing more than 50% null values. After deleting those columns, we only have one attribute with Null: '118. 8.9 - PRODUCTION RATIO'. For this attribute, we replaced the Null Values with median.

Variable Transformations

After conducting exploratory data analysis, we found that many variables in our dataset had high standard deviations, skewed distributions, or both. To account for these characteristics, multiple techniques were used such as Log Transformations and SMOTE were implemented.:

Synthetic Minority Oversampling Technique (SMOTE)

Since the target variable (the Cancer Rate Flag) for our dataset skewed more to cases of ‘0’, rather than ‘1’ (the Cancer Rate in the county based on the TRI report represents an extreme upper outlier), SMOTE was applied. SMOTE is a method used when dealing with imbalanced data in classification problems, where synthetic data points that are slightly different from the original data points are created, which creates an equal number of ‘NO’ and ‘YES’ values for our target variable.

USING SMOTE will both positively and negatively affect different metrics for assessing our model. Although SMOTE will cause an increase in recall, it will come at the cost of lower precision. In other words, SMOTE affects the model by reducing false negatives, at the expense of increasing false positives. As our target variable is concerned with detecting the presence of Carcinogens, which are cancer-causing compounds and as such is a matter of public health, having a model that prioritizes recall over precision is important.

Feature Selection

As there are many features in the dataset, the feature selection method using feature importances was implemented to choose the important features as some features are irrelevant or redundant for our analysis. Also, some of them are highly correlated to each other. Some of them are not providing any meaningful information. We want to drop those features. This reduction of features will decrease the computational complexity as well. After feature selection, the total number of attributes reduced to 122.

Primary Component Analysis (PCA)

We used primary component analysis (PCA) for dimensionality reduction. PCA transformed the original data with *n* number of features into *k* number of linear combinations of those original features where k <= n. If the *k* number of features provide as much information as the original data when *k* is less than *n*, we can just use those *k* number of linear combinations for our analysis. Here also, the computational complexity of our analysis will be reduced.

PCA provided an array of variances of each component in descending order. We have inspected those numbers by a scree plot. Also, we have considered the variances as the amount of information. We will drop those components providing less information (low variances).

The original dataset has 414 features. By looking at the Scree plot and the explained variance ratio of variance, we can find an elbow or an inflection point on the plot. We find that point around 130. After this point components are not giving us much information.

Comparing the classification reports of the original model and the model using PCA we see, by reducing the components from 414 to 130 (almost 1/3rd data) we are getting the scores near to the original model. By reducing the number of components, complexity of the model reduces. Without disturbing the performance of the original model much, PCA value 130 works good for analyzing this dataset.

Hyper Parameter Tuning

Hyperparameter plays an essential role in the fitting of supervised machine learning algorithms. However, it is computationally expensive to tune all the tunable hyperparameters simultaneously especially for large data sets.

Using Hyper Parameter tuning we are trying to control the learning process and determine the values of model parameters that a learning algorithm ends up learning. The prefix 'hyper_' suggests that they are 'top-level' parameters that control the learning process and the model parameters that result from it. While creating a model we start with the training process with random parameter values and adjust them throughout. Whereas, hyperparameters are the components set by before the training of the model.

Here we have used RandomizedSearchCV instead of GridSearchCV so that we can fix the number of parameter settings . First we have scaled our X variables[attributes].

We have tested the tuning for two models: neural Network and Random Forest, because we felt it is essential for the overall performance of the machine learning model before we finalized any model to start with.

Apart from the Neural Network, the Tree model is one of the key machine learning models of our experiment because of its robustness to noise, tolerance against missing information, handling of irrelevant, redundant predictive attribute values, low computational cost, interpretability, fast run time and robust predictors. To tune the decision tree, we tried hyperparameter tuning to check the range of values I should try for the maximum depth, which should be the minimum number of samples required at a leaf node.

Data Mining Models and Evaluations

In order to determine if a county has an elevated cancer rate, we first divided the dataset into four parts. First is the original dataset where none of the data preprocessing has been applied. Other three datasets have been obtained by applying data pre-processing methodologies like SMOTE (balancing the dataset), Feature Selection (which provides those features which are highly importance to decide whether a county has increased cancer rate) and PCA (which reduces the dimensionality by transforming the original data with *n* number of features into *k* number of linear combinations of those original features where k <= n).

Now, these four datasets have been used to build the following machine learning models:

Tree Based Models

Tree based algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based algorithms empower predictive models with high accuracy, stability and ease of interpretation. Unlike linear models, they map nonlinear relationships quite well. So we have created multiple tree algorithms to obtain good results.

Decision Tree

Decision Trees are a type of Supervised Machine Learning (that is you explain what the input is and what the corresponding output is in the training data) where the data is continuously split according to a certain parameter. The tree can be explained by two entities, namely decision nodes and leaves.

Random Forest

The random forest is a classification algorithm consisting of many decision trees. It uses bagging and features randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

Gradient Boosting

Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model in order to minimize the error.

Between all the tree models we obtained very good scores in terms of a metric for Gradient Boosting and Random Forest. The basic difference between these 2 models is - in Random forests, the results of decision trees are aggregated at the end of the process but in Gradient boosting doesn't do this and instead aggregates the results of each decision tree along the way to calculate the final result. The biggest advantage of decision trees is that they make it very easy to interpret and visualize nonlinear data patterns. Compared to other algorithms, decision trees require less effort for data preparation during pre-processing. A decision tree does not require normalization of data. A decision tree does not require scaling of data as well.

Naive Bayesian Classification

We have also created a model using Naive Bayesian algorithm by feeding all the dataset. Decision tree is a discriminative model, whereas Naive Bayes is a generative model. Decision trees are more flexible and easy. Naive Bayesian performs well in multi-class prediction when assumptions of independence are held.

Artificial Neural Network

In order to predict elevated cancer rates near Washington state facilities, the next model that we used was Artificial Neural Network. Artificial neural networks (ANN) are inspired by neural networks present in animal brains. Artificial neural networks consist of 3 stages of layers - input layer, hidden layers and output layer. Each layer in an ANN model consists of nodes which are interconnected using links or paths. Each connection between the nodes is associated with weights. If the weight on a path is positive the path is excitatory, otherwise it is inhibitory. Each neuron has a fixed threshold. If the net input into the neuron is greater than the input into the neuron is greater than the threshold, the neuron fires. This is the basic theory behind how the neuron works.

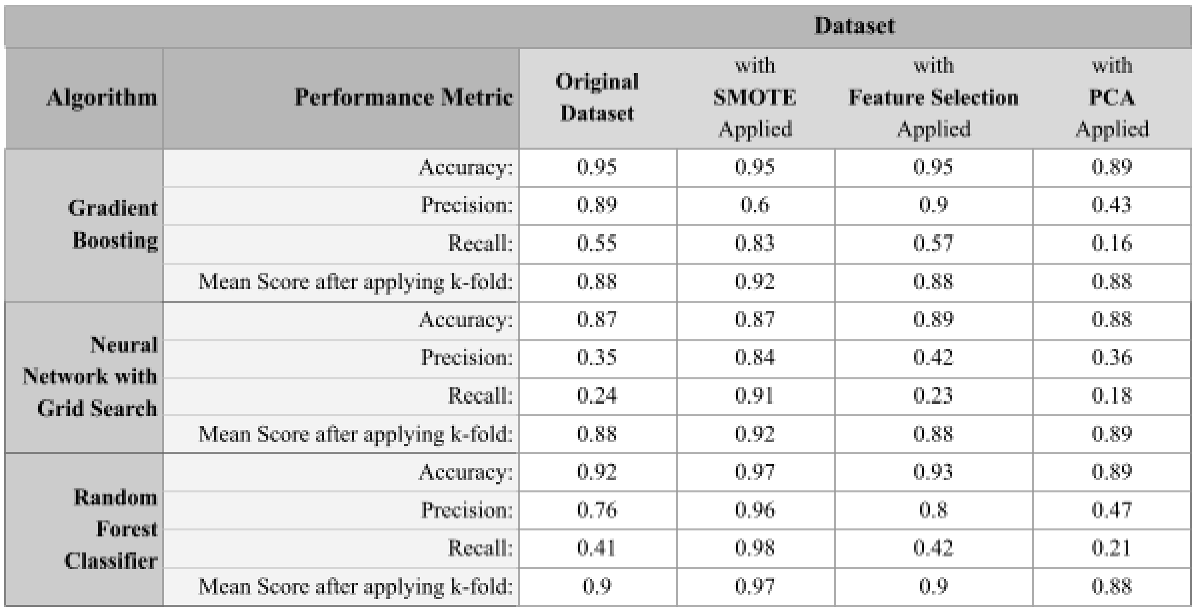
In this project, we have used a multilayer perceptron (MLP). It is a feedforward artificial neural network that generates a set of outputs from a set of inputs. 1 hidden layer of 3 neurons have been configured in the models. These hyperparameters are set before fitting the model using training data. The parameters defined during hyperparameter tuning are - number of hidden layers and in turn number of nodes in each hidden layer (hidden_layer_sizes), activation type (activation) and maximum iterations (max_iter).

Four different datasets have been fed for both GridsearchCV and RandomizedsearchCV. Once the data is loaded, all the predictor variables are normalized using Z-score normalization. Next, an MLPClassifier object is created. The beauty of GridsearchCV and RandomizedsearchCV is that they use K-fold cross validation by default.

Results

In the previous section, we discussed the different models that we have tried in our experiment to predict the cancer rate for counties. While considering the models we ran for each datasets, the outcomes are pretty consistent except Naive Bayesian Model. Because Naive Bayes assumes that all predictors (or features) are independent which our dataset hardly ensured. Although Neural network models are expensive in terms of computation time, we will suggest Neural Network with grid search as all of the other models are Tree based whereas Neural Network helped computers make intelligent decisions and gave good results. After checking the scores for all the models, we have come up with the below 3 models which are performing great for our experiment. We will be suggesting Gradient boosting by seeing the recall value as we are classifying cancer rate as our target variable and this metric is really important. And to check the robustness of the models , the other models can be used.

Figure 14: Comparison of Performance Metrics of Top Three Machine Learning Algorithms Tested



		Dataset			
Algorithm	**Performance Metric**	**Original Dataset**	with **SMOTE** Applied	with **Feature Selection** Applied	with **PCA** Applied
Gradient Boosting	Accuracy:	0.95	0.95	0.95	0.89
	Precision:	0.89	0.6	0.9	0.43
	Recall:	0.55	0.83	0.57	0.16
	Mean Score after applying k-fold:	0.88	0.92	0.88	0.88
Neural Network with Grid Search	Accuracy:	0.87	0.87	0.89	0.88
	Precision:	0.35	0.84	0.42	0.36
	Recall:	0.24	0.91	0.23	0.18
	Mean Score after applying k-fold:	0.88	0.92	0.88	0.89
Random Forest Classifier	Accuracy:	0.92	0.97	0.93	0.89
	Precision:	0.76	0.96	0.8	0.47
	Recall:	0.41	0.98	0.42	0.21
	Mean Score after applying k-fold:	0.9	0.97	0.9	0.88

<u>Discussion</u>

Domain Knowledge

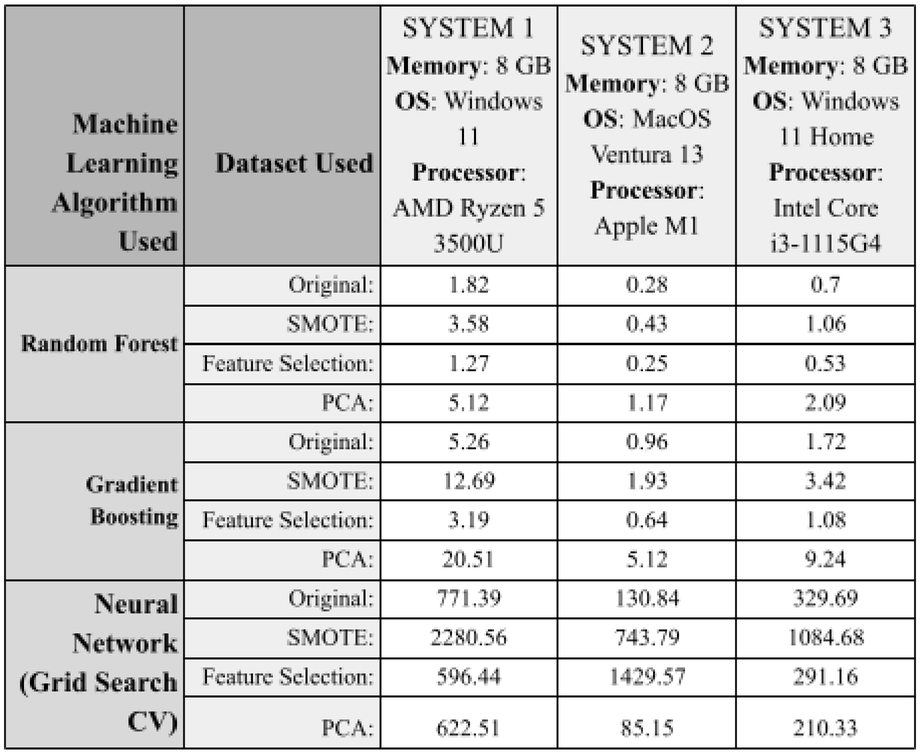
We started with the problem statement "Predicting the presence of carcinogens (cancer-causing compounds) in toxic releases from Washington state industrial & federal facilities". We started with the TRI report for 2021 and performed all data pre processing. At this stage, we felt that we should merge the TRI dataset with another dataset to take our project to a new direction. We added a new dataset from the Cancer Registry of Washington State Department of Health, which tracks incidents of cancer across counties as well as that county's estimated annual population, from 1994 to 2018. We merged these two datasets by performing an inner join on year & county. Also, rather than focusing on TRI reports from a single year, we've added many more years into our dataset. This new dataset became size wise 25x larger than our old dataset.

Although our second dataset from the Washington State Cancer Registry contained numeric values for Cancer and Population in individual years and counties, we made the decision to transform this dataset so it could be implemented as a classification problem rather than a regression problem. Our reasons for doing so are two-fold.

The first is that there are many different factors which can impact the rates of cancer in a given county and it's difficult to assess how much these factors can be controlled for in a model. For example, if a county has a higher rate of smokers, then it's likely that there will be a higher incidence of cancer in that county as well. While a factor such as this might be more easily controlled, there are others such as say an environmental incident that has contaminated the local water supply which are more difficult to control for. There are many unknown factors within this example that could have an effect on cancer rates such as: How prolonged was the water contaminated? From the source of the contamination, how far did the contamination spread? The Oncology (the study of Cancer) industry is valued at a quarter of a trillion dollars (Precedence Research) and there is still much work to be done for understanding the many different factors which led to Cancer.

The second is that we wanted to create a metric which not just provided clearer comprehension for stakeholders, while taking into account the gap of knowledge humans have in understanding and measuring the impact of Cancer. Setting our threshold for the cancer rate flag at the level of an upper range extreme outlier hopefully accounts for the high level of variance inherent in our limited, though growing, understanding of this grave disease. By setting this threshold so high, it gives relevant stakeholders a clear metric that a factory is worthy of investigating to make sure there aren't dangerous levels of output occuring that harm the communities they exist in.

Figure 14: Seconds to Compute For ML Algorithms



Machine Learning Algorithm Used	**Dataset Used**	SYSTEM 1 **Memory**: 8 GB **OS**: Windows 11 **Processor**: AMD Ryzen 5 3500U	SYSTEM 2 **Memory**: 8 GB **OS**: MacOS Ventura 13 **Processor**: Apple M1	SYSTEM 3 **Memory**: 8 GB **OS**: Windows 11 Home **Processor**: Intel Core i3-1115G4
Random Forest	Original:	1.82	0.28	0.7
	SMOTE:	3.58	0.43	1.06
	Feature Selection:	1.27	0.25	0.53
	PCA:	5.12	1.17	2.09
Gradient Boosting	Original:	5.26	0.96	1.72
	SMOTE:	12.69	1.93	3.42
	Feature Selection:	3.19	0.64	1.08
	PCA:	20.51	5.12	9.24
Neural Network (Grid Search CV)	Original:	771.39	130.84	329.69
	SMOTE:	2280.56	743.79	1084.68
	Feature Selection:	596.44	1429.57	291.16
	PCA:	622.51	85.15	210.33

As it stands, this model provides a framework for predicting power plants that are a cause of concern for the counties in which they operate and provides an easily comprehensible tool for stakeholders to identify such a concern.

Methodological Contributions

With the combined dataset, we proceeded with hyperparameter tuning and then built five machine learning models- Decision Tree, Random Forest, Gradient Boosting, Naive Bayesian Classification and Neural Network model. At first, we were getting abnormally very high accuracy (99%-100%) for all the tree models (Decision Tree, Random Forest and Gradient Boosting), which was weird. This could be due to overfitting issues or due to high correlation between variables. We built the correlation matrix again after Primary Component Analysis and inspected Multicollinearity. At this point we got rid of all the highly correlated variable pairs. Also, we inspected the GINI importance of each feature. We found a feature where the GINI importance was a lot higher than anything else. Once we drop that, performance metrics seem more believable.

Our research shows that if a county has its toxic release data and cancer data across counties as well as that counties' estimated annual population data, our machine learning model can predict if that county has an elevated cancer rate. Knowing this prior will help policy makers and the health department plan accordingly. For example, policymakers can have stricter industrial regulations in a county with elevated cancer risks. Also, counties with a higher rate of cancer possibility can reduce exposure to harmful environment factors by maintaining strict guidelines issued by the health department.

This model can be used by policy makers or EPA as it can determine if a county has an elevated cancer rate or not. Nowadays, society is becoming more aware of the negative environmental impact of factories, impact of global warming, etc. Fines can be used as a way to effect change for the upcoming days. We can take our analysis to a next level which could be more beneficial to stakeholders by equipping IOT devices at factories which take the numerical inputs and measure the levels of outputs. This can allow the stakeholders or city planners to keep things under control. The model can be used by factories as well. As the measurement metrics become more accurate, staying in compliance to avoid penalties whether they be fines, shutdowns or lawsuits is of utmost importance. The models could be used to help them avoid these.

Conclusions

Summary

Our research focuses on prediction if a county has an elevated cancer rate as indicated by data provided by Toxic Release Inventory on Washington State Industrial Facilities. Our research uses various machine learning models which could be implemented by the government environmental or health departments. Throughout the process we have also explained which models performed better than others and the reason behind that. Out of all models Random Forest, Gradient Boosting, Neural Network w/ Grid Search acted best. There were some limitations which we are discussing below.

Limitations

A primary limitation of our study is with regards to the nature of cancer itself. In our study, there is a false premise created for the sake of creating a model that cancer is only caused by a factory's output from that year, with no regards to cancer's nature as a disease which can reveal itself over a much longer timespan, be it months, years, or decades. For example, if someone smokes when they are younger, even if they quit, they will nonetheless have a higher possibility of being diagnosed with lung cancer over the course of their lifetime than someone who had never smoked (John Hopkins). In other words, when cancer reveals itself in a living organism varies wildly and the length of and intensity of exposure to carcinogens have a dramatic effect. In the context of our study, someone who lives in the same county as a factory for which the Cancer Rate Flag was triggered might not be diagnosed for cancer in the year in the year they live next to the factory, but living in the same county as such a factory could cause them to have a higher risk of cancer which would only reveal itself decades later.

There are also many other causes of cancer which aren't taken into account in the model. For example, our model does not include the percentage of smokers in a county or a county's average age. How much someone smokes and how old someone has been shown to be strongly correlated with cancer (Center for Disease Control). Other demographic factors such as race are not accounted for, which has been shown to have an impact on rates of screening, treatment and overall outcomes (Kaiser Family Foundation).

Additionally, our model doesn't take into account multiple TRI reports being listed for a given county. Although King County might have more TRI reports than any other county, not only is the number of reports in a given county not taken into account, but also the grade of specific measurements within the report. For example, a smaller facility may have lower output than a larger industrial facility, so with that in mind the level of an output in a certain facility should be measured as well, which our current model does not consider. Aggregating multiple reports, as well as being able to weigh individual facility's level of output, might yield better results with regards to predicting Cancer Rates.

Currently the distance metric utilized is imprecise. In terms of overall area, some counties are much larger than others. Additionally, we aren't able to see individual level data on those who are diagnosed with cancer, so we can't assess individual's proximity to industrial facilities. Healthcare data is anonymized so it makes it difficult to do a more tightly controlled study on a factory's impacts on cancer rates.

Additionally, our model being based on an upper range extreme outlier, does not account for potential state-wide or greater changes in cancer rates. For example, the radioactive fallout from the Chernobyl accident went beyond immediate manmade-delineated borders (National Library of Medicine). Regardless of the root cause for an increase in cancer rates, the model wouldn't be able to account and yet the threshold would be raised.

Lastly, having Oncologists as well as environmental scientists directly involved with the implementation and modifications of the model would provide a more solid scientific foundation for this study. Domain experts would undoubtedly provide better context and may help prevent unintended consequences regarding the implementation of this model.

Future Projects

Although this project provides a foundational framework of a tool that can be used to better understand the impact of factories on the communities they operate in, there is a great opportunity for improvement that would address some of the limitations previously addressed.

Implementing missing features that show a clear casual relationship with cancer such as the average age and the smoking rates in counties for that year would improve the model. Other demographic factors such as the number of factory workers in a county may improve the model.

In addition, adding features which account for other TRI reports in a given county for that year might yield an improved model. As discussed in limitations, someone may be exposed to carcinogens earlier in their life which may increase their risk of cancer later in life. Adding time series elements regarding factory output from previous years may address some of these shortcomings.

County is an artificially defined border whereas cancer is a phenomena that doesn't pay attention to borders. Existing zoning regulations affect where factories may or may not operate. This model may see improvement through using longitude and latitude of a factory's location to the most densely populated point in a given county would be improvement towards addressing factory location relative to residential areas in a county.

In the results, we obtained high accuracy scores in most of the Tree models, specially with SMOTE dataset, which could indicate an overfitting problem. In future, we can solve the problem of overfitting by: Increasing the training data by data augmentation. If we find a way to reduce the complexity, then the overfitting issue can be overcome. We have set the cancer prediction 'YES' if the value is greater than equal to the extreme outliers as we didnt have any valid data to fix the threshold to segregate. So we can work further to fix this threshold in future for better prediction.

Lastly, we can extend the research to determine the impact of releases by facilities which contain carcinogen in air and in ground. This way we can gauge the impact on respiratory diseases which are caused by air borne releases and impact on food / agriculture which in turn are caused by water or ground releases.

References

1. Nikhil Agarwal, Chanont Banternghansa, Linda T.M. Bui. (2010). Toxic exposure in America:Estimating fetal and infant health outcomes from 14 years of TRI reporting. [Online]. URL Toxic exposure in America: Estimating fetal and infant health outcomes from 14 years of TRI reporting

2. Thomas P. McKeon & Wei-Ting Hwang & Zhuoran Ding & Vicky Tam & Paul Wileyto & Karen Glanz & Trevor M. Penning. (2021). Environmental exposomics and lung cancer risk assessment in the Philadelphia metropolitan area using ZIP code–level hazard indices. [Online]. URL Environmental exposomics and lung cancer risk assessment in the Philadelphia metropolitan area using ZIP code–level hazard indices

3. S.M.Lee1D.L.Hershman , P.Martin J.P.Leonard &Y.K.Cheung (March,2011).Toxicity burden score: a novel approach to summarize multiple toxic effects. [Online]. URL Toxicity burden score: a novel approach to summarize multiple toxic effects

4. Irene Jacqz (September 2022). Toxic test scores: The impact of chemical releases on standardized test performance within U.S. schools.[Online]. URL Toxic test scores: The impact of chemical releases on standardized test performance within U.S. schools

5. Wilson, S. M., Fraser-Rahim, H., Williams, E., Zhang, H., Rice, L., Svendsen, E., & Abara, W. (2012). Assessment of the distribution of toxic release inventory facilities in metropolitan Charleston: an environmental justice case study. *American journal of public health*, *102*(10), 1974–1980. https://doi.org/10.2105/AJPH.2012.300700
6. Laufer, B. (2018, January). *Data Analytics and Machine Learning for Environmental Protection: Targeting Air Inspections*. Retrieved October 9, 2022, from https://bendlaufer.github.io/Laufer_Benjamin_EPA_Project.pdf
7. Princeton University, Department of Operations Research and Financial Engineering Volunteer Intern, EPA Region 2 DECA–CAPSB, Data Management Team
8. Hendryx, Luo, J., & Chen, B.-C. (2014). Total and cardiovascular mortality rates in relation to discharges from toxics release inventory sites in the United States. Environmental Research, 133, 36–41. https://doi.org/10.1016/j.envres.2014.05.010
9. Léa Fortunato, Juan José Abellan, Linda Beale, Sam LeFevre, Sylvia Richardson. (2011). Spatio-temporal patterns of bladder cancer incidence in Utah (1973-2004) and their association with the presence of toxic release inventory sites. [Online]. URL https://ij-healthgeographics.biomedcentral.com/articles/10.1186/1476-072X-10-16
10. Pennacchiotti, M., & Popescu, A.-M. (2011). A Machine Learning Approach to Twitter User Classification. Proceedings of the International AAAI Conference on Web and Social Media, 5(1), 281–288. https://ojs.aaai.org/index.php/ICWSM/article/view/14139/13988
11. Chauhan, H., & Chauhan, A. (2013). Implementation of decision tree algorithm c4.5. International Journal of Scientific and Research Publications, 3(10). https://citeseerx.ist.psu.edu/viewdoc/download? doi=10.1.1.395.9610&rep=rep1&type=pdf
12. Roser M, Ritchie H. Cancer. (2017, April). Our World in Data. https://ourworldindata.org/cancer#:~:text=Global%20cancer%20prevalence%20has%20risen,(largely%20due%20to%20smoking).
13. Racial Disparities in Cancer Outcomes, Screening, and Treatment. (2022b, February 3). KFF. https://www.kff.org/racial-equity-and-health-policy/issue-brief/racial-disparities-in-cancer-outcomes-screening-and-treatment/
14. Former Smokers: What's Your Risk for Lung Cancer? (2021, August 8). Johns Hopkins Medicine. https://www.hopkinsmedicine.org/health/conditions-and-diseases/lung-cancer/former-smoker-whats-your-risk-for-lung-cancer

15. Cardis E, Krewski D, Boniol M, et al. Estimates of the cancer burden in Europe from radioactive fallout from the Chernobyl accident. Int J Cancer. 2006;119(6):1224-1235. doi:10.1002/ijc.22037

Data Sources

2021 TRI Preliminary Dataset: Basic Data Files. (2022, September 28). US EPA. Retrieved October 1, 2022, from [2021 TRI Preliminary Dataset: Basic Data Files | US EPA].

1995 - 2018 Washington State Cancer Registry. (2022, June). Retrieved November 6, 2022, from [Washington State Department of Health].

Appendix

FIGURE A: Univariate Analysis on Numeric Variables

Feature	Standard Dev	Mean	max	25%	50%	75%
5.1 - FUGITIVE AIR	1,599,292.69	178,603.87	43,482,000	33.60	2,296	32,942
5.2 - STACK AIR	1,368,595.06	56,972.12	43,482,000	0.00	0	0
5.3 - WATER	643,313.24	48,661.27	19,267,609	0.00	12	2,046
5.5.1B - OTHER LANDFILLS	536,190.35	30,201.42	15,828,830	0.00	0	0
5.5.2 - LAND TREATMENT	536,190.35	30,201.42	15,828,830	0.00	0	0
5.5.3B - OTHER SURFACE I	473,357.47	55,721.30	8,160,000	0.00	0	0
5.5.4 - OTHER DISPOSAL	473,083.88	15,461.67	15,225,837	0.00	0	0
ON-SITE RELEASE TOTAL	209,055.31	10,558.23	6,601,328	0.00	0	0
6.1 - POTW - TRNS RLSE	170,592.64	21,022.42	3,745,995	3.00	201	3,450
6.1 - POTW - TRNS TRT	125,341.11	12,185.82	3,745,995	0.03	30	985
POTW - TOTAL TRANSFERS	124,819.55	11,589.75	3,745,995	0.03	29	928
6.2 - M10	115,919.77	8,836.60	3,438,779	0.00	0	26
6.2 - M41	111,135.13	6,554.81	3,433,471	0.00	0	0
6.2 - M62	106,672.34	3,352.69	3,433,470	0.00	0	0

6.2 - M64	100,235.16	4,629.30	3,151,982	0.00	0	0
6.2 - M65	86,828.58	5,587.94	2,641,443	0.00	0	0
6.2 - M65	61,245.95	6,393.16	1,374,306	0.00	0	0
6.2 - M73	61,245.93	6,393.36	1,374,306	0.00	0	0
6.2 - M79	55,982.54	3,945.76	1,374,306	0.00	0	0
6.2 - M90	53,441.84	3,440.40	1,374,306	0.00	0	0
6.2 - M94	41,962.56	1,883.27	1,107,155	0.00	0	0
6.2 - M99	40,264.50	6,853.76	313,056	0.00	0	0
OFF-SITE RELEASE TOTAL	36,886.07	3,213.51	806,379	0.00	0	0
6.2 - M20	36,885.87	3,229.89	806,379	0.00	0	0
6.2 - M24	36,885.87	3,229.89	806,379	0.00	0	0
6.2 - M26	32,788.26	2,275.94	965,721	0.00	0	0
OFF-SITE RECYCLED TOTAL	28,274.61	4,456.55	505,922	0.00	3	244
6.2 - M56	24,702.55	2,298.26	602,993	0.00	0	0
OFF-SITE ENERGY RECOVERY T	22,388.74	1,495.69	502,616	0.00	0	0
6.2 - M40 NON-METAL	18,819.14	797.77	594,013	0.00	0	0
6.2 - M50	13,208.27	1,257.14	234,177	0.00	0	0
6.2 - M54	12,710.28	1,484.93	363,103	0.00	0	51
6.2 - M61 NON-METAL	11,867.09	892.79	243,646	0.00	0	0
6.2 - M69	11,752.96	708.42	257,002	0.00	0	0
6.2 - M95	11,729.49	590.71	313,056	0.00	0	0
OFF-SITE TREATED TOTAL	7,048.05	505.36	184,550	0.00	0	0
6.2 - TOTAL TRANSFER	3,652.70	210.94	92,681	0.00	0	0
TOTAL RELEASES	3,156.67	135.70	85,901	0.00	0	0
8.1A - ON-SITE CONTAINED	2,906.35	111.62	90,081	0.00	0	0
8.1B - ON-SITE OTHER	2,300.40	98.12	65,164	0.00	0	0
8.1C - OFF-SITE CONTAIN	2,222.57	219.98	38,841	0.00	0	0
8.1D - OFF-SITE OTHER R	2,222.57	219.98	38,841	0.00	0	0

8.2 - ENERGY RECOVER ON	1,808.59	70.41	56,734	0.00	0	0
8.3 - ENERGY RECOVER OF	1,495.08	93.19	32,677	0.00	0	0
8.4 - RECYCLING ON SITE	1,262.50	77.55	32,969	0.00	0	0
8.5 - RECYCLING OFF SITE	488.40	27.72	12,867	0.00	0	0
8.6 - TREATMENT ON SITE	301.47	16.38	8,350	0.00	0	0
8.7 - TREATMENT OFF SITE	171.90	15.80	3,291	0.00	0	0
PRODUCTION WSTE (8.1-8.7)	148.93	10.97	3,918	0.00	0	0
8.8 - ONE-TIME RELEASE	12.25	1.61	374	0.95	1	1
8.9 - PRODUCTION RATIO	0.31	0.01	10	0.00	0	0

Note: The minimum value for all columns is 0. The count for all columns is 1036, with the exception of ‘6.2 - M99’ which is 106, and ‘8.8 - ONE-TIME RELEASE’ which is 987.

**** See ‘Numerical Data Description (Pre-Processing)’ in Appendix**

FIGURE B: Correlation Heatmap

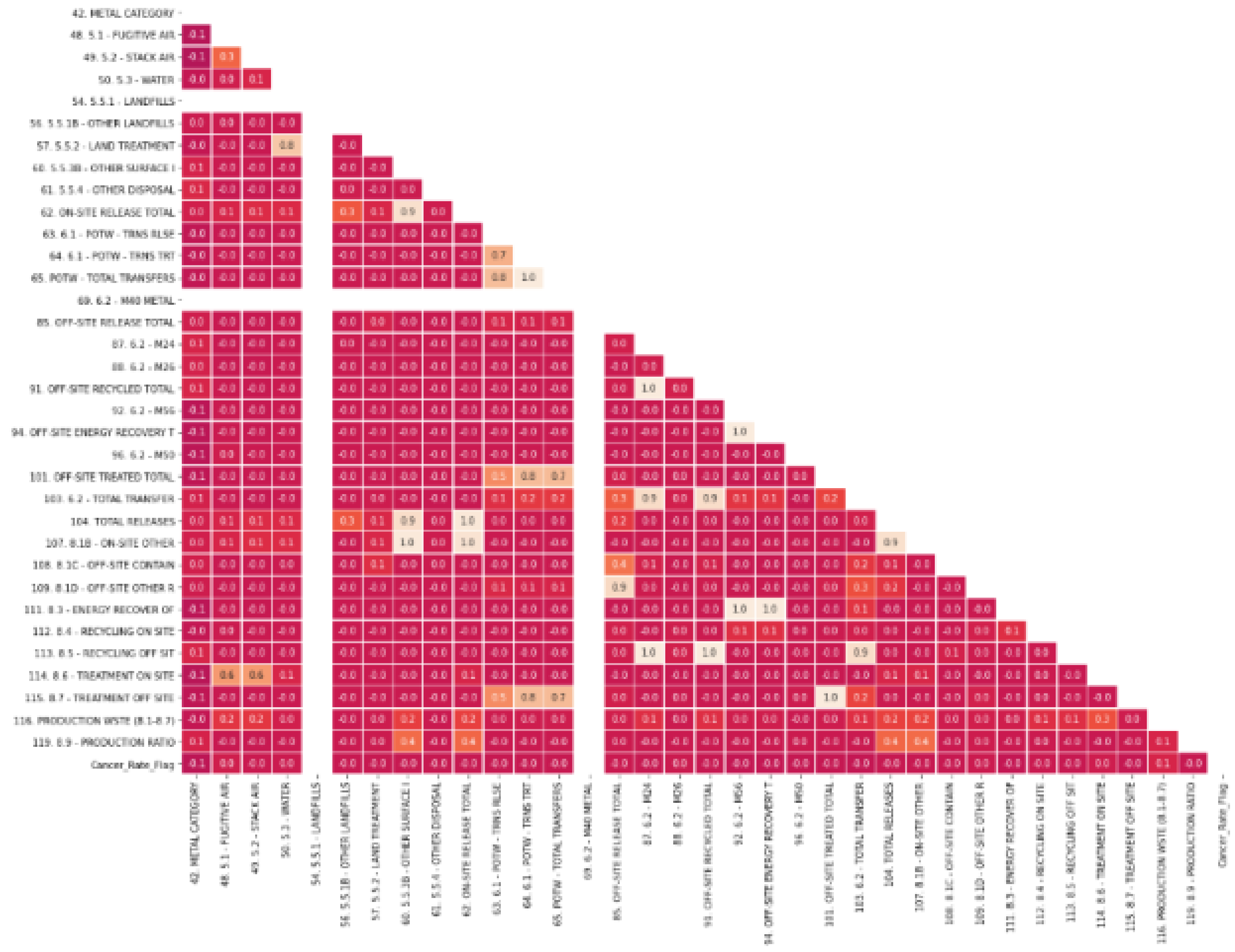


FIGURE C: Chemical Frequency in Toxic Release Inventory Reports

Chemical	% Count
Lead compounds	5.36%
Lead	5.16%
Nitrate Compounds	4.02%
Ammonia	3.41%
Xylene (mixed isomers)	2.99%
Toluene	2.91%
Methanol	2.89%
Nitric acid	2.77%
Styrene	2.70%
Copper	2.66%
Polycyclic aromatic compounds	2.62%
Manganese	2.33%
Chromium	2.24%
Manganese compounds	1.98%
Zinc compounds	1.96%
Naphthalene	1.86%
Nickel	1.81%
Dioxin and dioxin-like compounds	1.71%
Chromium compounds (except for chromite ore mined in the Transvaal Region)	1.70%
Ethylbenzene	1.70%
Copper compounds	1.67%
Benzo[g,h,i]perylene	1.65%
Chlorine	1.59%
1,2,4-Trimethylbenzene	1.59%
Certain glycol ethers	1.53%
Phenol	1.52%
Hydrochloric acid (acid aerosols including mists, vapors, gas, fog, and other airborne forms of any particle size)	1.52%
Hydrogen fluoride	1.49%
Benzene	1.41%
Diisocyanates	1.40%

n-Hexane	1.39%
Mercury compounds	1.34%
Formaldehyde	1.13%
Ethylene glycol	1.05%
Sulfuric acid (acid aerosols including mists, vapors, gas, fog, and other airborne forms of any particle size)	1.00%
Methyl ethyl ketone	0.99%
n-Butyl alcohol	0.94%
Nickel compounds	0.83%
Methyl isobutyl ketone	0.81%
Acetaldehyde	0.78%
Barium compounds (except for barium sulfate (CAS No. 7727-43-7))	0.74%
Cyclohexane	0.73%
Phosphoric acid	0.69%
Formic acid	0.61%
Catechol	0.58%
Propylene	0.58%
Diethanolamine	0.57%
Chlorine dioxide	0.56%
Cresol (mixed isomers)	0.55%
Carbonyl sulfide	0.53%
Tetrachloroethylene	0.50%
Arsenic compounds	0.50%
Ethylene	0.48%
Dichloromethane	0.44%
Trichloroethylene	0.43%
Cumene	0.41%
1,3-Butadiene	0.40%
Pentachlorophenol	0.40%
Carbon disulfide	0.37%
Molybdenum trioxide	0.37%
Hydrogen sulfide	0.37%
N-Methyl-2-pyrrolidone	0.32%
Mercury	0.29%
Phenanthrene	0.29%

Aluminum (fume or dust)	0.28%
Di(2-ethylhexyl) phthalate	0.25%
Cobalt compounds	0.25%
Chloroform	0.25%
Methyl methacrylate	0.23%
Tetrabromobisphenol A	0.22%
Biphenyl	0.22%
Antimony compounds	0.22%
Metham sodium	0.21%
Anthracene	0.20%
Hydrogen cyanide	0.19%
Polychlorinated biphenyls	0.19%
Toluene diisocyanate (mixed isomers)	0.18%
Lithium carbonate	0.17%
Zinc (fume or dust)	0.17%
Chlorodifluoromethane (HCFC-22)	0.17%
Cyanide compounds	0.15%
Dimethylamine	0.15%
Dimethyl phthalate	0.14%
Silver	0.14%
Methyl tert-butyl ether	0.14%
Sodium dimethyldithiocarbamate	0.14%
Sodium nitrite	0.14%
Hexachlorobenzene	0.12%
Acrylic acid	0.12%
Antimony	0.12%
Vanadium compounds	0.11%
Acetonitrile	0.11%
Lead And Lead Compounds	0.11%
Cadmium compounds	0.10%
Creosote	0.10%
N,N-Dimethylformamide	0.09%
Potassium dimethyldithiocarbamate	0.09%
Freon 113 (CFC-113)	0.09%

Diphenylamine	0.09%
1,1,1-Trichloroethane	0.08%
Peracetic acid	0.08%
Acrylamide	0.08%
sec-Butyl alcohol	0.08%
Benzoyl peroxide	0.08%
1,3-Dichloropropylene	0.08%
4,4'-Isopropylidenediphenol	0.08%
1,2-Dichloroethane	0.07%
1,1-Dichloro-1-fluoroethane (HCFC-141b)	0.07%
Chloropicrin	0.07%
trans-1,3-Dichloropropene	0.07%
Nitroglycerin	0.07%
Pyridine	0.07%
Potassium N-methyldithiocarbamate	0.06%
Mixture	0.06%
Asbestos (friable)	0.06%
Dibutyl phthalate	0.06%
Dichlorodifluoromethane (CFC-12)	0.05%
Chloromethane	0.05%
Decabromodiphenyl oxide	0.05%
Chromium and Chromium Compounds(except for chromite ore mined in the Transvaal Region)	0.04%
Triethylamine	0.04%
Phosphorus (yellow or white)	0.04%
Copper And Copper Compounds	0.04%
Propionaldehyde	0.04%
1-Bromopropane	0.03%
Cobalt	0.03%
Diuron	0.03%
Selenium compounds	0.03%
Vinyl acetate	0.03%
Aluminum oxide (fibrous forms)	0.03%
Dinitrotoluene (mixed isomers)	0.03%
Isoprene	0.03%

Manganese And Manganese Compounds	0.02%
Nickel And Nickel Compounds	0.02%
Propylene oxide	0.02%
Propiconazole	0.02%
Mercury And Mercury Compounds	0.02%
Arsenic	0.02%
tert-Butyl alcohol	0.02%
Barium	0.02%
2-Methoxyethanol	0.02%
Barium And Barium Compounds	0.02%
Sodium azide	0.02%
Polychlorinated alkanes (C10-C13)	0.02%
Hydroquinone	0.01%
Isopropyl alcohol (only persons who manufacture by the strong acid process are subject, no supplier notification)	0.01%
Toluene-2,6-diisocyanate	0.01%
Quinoline	0.01%
Cadmium	0.01%
Nabam	0.01%
Toluene-2,4-diisocyanate	0.01%
Trichlorofluoromethane (CFC-11)	0.01%
Dicyclopentadiene	0.01%
2,4-Dimethylphenol	0.01%
m-Xylene	0.01%
Fluorine	0.01%
Acetophenone	0.01%
Cobalt And Cobalt Compounds	0.01%
1,2,4-Trichlorobenzene	0.01%
Ozone	0.01%
Phosphine	0.01%
1,4-Dioxane	0.01%
Butyl acrylate	0.01%
Disodium cyanodithioimidocarbonate	0.01%
Bromoxynil	0.00%
Trifluralin	0.00%

Triallate	0.00%
1,2-Dibromoethane	0.00%
Trade Secret	0.00%
Cyclohexanol	0.00%
2,2-Dichloro-1,1,1-trifluoroethane (HCFC-123)	0.00%
Dibenzofuran	0.00%
Carbon tetrachloride	0.00%
Folpet	0.00%
2-Phenylphenol	0.00%
Dichlorotetrafluoroethane (CFC-114)	0.00%
m-Cresol	0.00%
Nitrilotriacetic acid	0.00%
Dimethyl sulfate	0.00%
p-Xylene	0.00%
Vanadium (except when contained in an alloy)	0.00%
o-Xylene	0.00%

FIGURE D: Correlation Coefficients of Numeric Data

Feature 1	Feature 2	Correlation Coefficient
103. 6.2 - TOTAL TRANSFER	91. OFF-SITE RECYCLED TOTAL	0.999333846
91. OFF-SITE RECYCLED TOTAL	87. 6.2 - M24	0.999078445
64. 6.1 - POTW - TRNS TRT	65. POTW - TOTAL TRANSFERS	0.998993271
103. 6.2 - TOTAL TRANSFER	87. 6.2 - M24	0.998404315
115. 8.7 - TREATMENT OFF SITE	101. OFF-SITE TREATED TOTAL	0.986947184
104. TOTAL RELEASES	62. ON-SITE RELEASE TOTAL	0.972109373
60. 5.5.3B - OTHER SURFACE I	107. 8.1B - ON-SITE OTHER	0.964855990
94. OFF-SITE ENERGY RECOVERY T	92. 6.2 - M56	0.961081506
101. OFF-SITE TREATED TOTAL	64. 6.1 - POTW - TRNS TRT	0.941530632
65. POTW - TOTAL TRANSFERS	101. OFF-SITE TREATED TOTAL	0.940587329
64. 6.1 - POTW - TRNS TRT	115. 8.7 - TREATMENT OFF SITE	0.932733094

115. 8.7 - TREATMENT OFF SITE	65. POTW - TOTAL TRANSFERS	0.932048677
107. 8.1B - ON-SITE OTHER	62. ON-SITE RELEASE TOTAL	0.891730742
107. 8.1B - ON-SITE OTHER	104. TOTAL RELEASES	0.865366172
62. ON-SITE RELEASE TOTAL	60. 5.5.3B - OTHER SURFACE I	0.861843919
60. 5.5.3B - OTHER SURFACE I	104. TOTAL RELEASES	0.836583213
114. 8.6 - TREATMENT ON SITE	116. PRODUCTION WSTE (8.1-8.7)	0.635270697
49. 5.2 - STACK AIR	114. 8.6 - TREATMENT ON SITE	0.474910116
109. 8.1D - OFF-SITE OTHER R	85. OFF-SITE RELEASE TOTAL	0.434170476
108. 8.1C - OFF-SITE CONTAIN	85. OFF-SITE RELEASE TOTAL	0.405820403
62. ON-SITE RELEASE TOTAL	49. 5.2 - STACK AIR	0.350022077
49. 5.2 - STACK AIR	104. TOTAL RELEASES	0.338526048
49. 5.2 - STACK AIR	116. PRODUCTION WSTE (8.1-8.7)	0.313051817
56. 5.5.1B - OTHER LANDFILLS	62. ON-SITE RELEASE TOTAL	0.290793604
63. 6.1 - POTW - TRNS RLSE	65. POTW - TOTAL TRANSFERS	0.288020266
104. TOTAL RELEASES	56. 5.5.1B - OTHER LANDFILLS	0.283026601
62. ON-SITE RELEASE TOTAL	116. PRODUCTION WSTE (8.1-8.7)	0.277899442
116. PRODUCTION WSTE (8.1-8.7)	104. TOTAL RELEASES	0.273620913
48. 5.1 - FUGITIVE AIR	114. 8.6 - TREATMENT ON SITE	0.273481844
64. 6.1 - POTW - TRNS TRT	63. 6.1 - POTW - TRNS RLSE	0.244771018
112. 8.4 - RECYCLING ON SITE	116. PRODUCTION WSTE (8.1-8.7)	0.240562599
85. OFF-SITE RELEASE TOTAL	104. TOTAL RELEASES	0.240154617
107. 8.1B - ON-SITE OTHER	116. PRODUCTION WSTE (8.1-8.7)	0.237830156
115. 8.7 - TREATMENT OFF SITE	63. 6.1 - POTW - TRNS RLSE	0.233808627
101. OFF-SITE TREATED TOTAL	63. 6.1 - POTW - TRNS RLSE	0.230558015
57. 5.5.2 - LAND TREATMENT	50. 5.3 - WATER	0.227669557
62. ON-SITE RELEASE TOTAL	61. 5.5.4 - OTHER DISPOSAL	0.206572779
104. TOTAL RELEASES	61. 5.5.4 - OTHER DISPOSAL	0.204033630
114. 8.6 - TREATMENT ON SITE	62. ON-SITE RELEASE TOTAL	0.192428252
48. 5.1 - FUGITIVE AIR	49. 5.2 - STACK AIR	0.187697134

104. TOTAL RELEASES	114. 8.6 - TREATMENT ON SITE	0.184778977
48. 5.1 - FUGITIVE AIR	116. PRODUCTION WSTE (8.1-8.7)	0.183986117
50. 5.3 - WATER	114. 8.6 - TREATMENT ON SITE	0.172733238
116. PRODUCTION WSTE (8.1-8.7)	113. 8.5 - RECYCLING OFF SIT	0.170566389
60. 5.5.3B - OTHER SURFACE I	116. PRODUCTION WSTE (8.1-8.7)	0.169075704
62. ON-SITE RELEASE TOTAL	50. 5.3 - WATER	0.163131193
61. 5.5.4 - OTHER DISPOSAL	107. 8.1B - ON-SITE OTHER	0.161452930
104. TOTAL RELEASES	50. 5.3 - WATER	0.159765811
62. ON-SITE RELEASE TOTAL	48. 5.1 - FUGITIVE AIR	0.144192571
48. 5.1 - FUGITIVE AIR	104. TOTAL RELEASES	0.139121076
50. 5.3 - WATER	116. PRODUCTION WSTE (8.1-8.7)	0.131237188
57. 5.5.2 - LAND TREATMENT	108. 8.1C - OFF-SITE CONTAIN	0.129452101
60. 5.5.3B - OTHER SURFACE I	61. 5.5.4 - OTHER DISPOSAL	0.123893667
49. 5.2 - STACK AIR	50. 5.3 - WATER	0.120468480
57. 5.5.2 - LAND TREATMENT	85. OFF-SITE RELEASE TOTAL	0.115529397
107. 8.1B - ON-SITE OTHER	49. 5.2 - STACK AIR	0.114321835
50. 5.3 - WATER	107. 8.1B - ON-SITE OTHER	0.106896118
104. TOTAL RELEASES	109. 8.1D - OFF-SITE OTHER R	0.103890743
61. 5.5.4 - OTHER DISPOSAL	42. METAL CATEGORY	0.099276127
114. 8.6 - TREATMENT ON SITE	107. 8.1B - ON-SITE OTHER	0.098683497
114. 8.6 - TREATMENT ON SITE	54. 5.5.1 - LANDFILLS	0.097927046
109. 8.1D - OFF-SITE OTHER R	63. 6.1 - POTW - TRNS RLSE	0.096769545
49. 5.2 - STACK AIR	42. METAL CATEGORY	0.096470387
104. TOTAL RELEASES	108. 8.1C - OFF-SITE CONTAIN	0.093746587
85. OFF-SITE RELEASE TOTAL	69. 6.2 - M40 METAL	0.091863108
113. 8.5 - RECYCLING OFF SIT	103. 6.2 - TOTAL TRANSFER	0.088070036
113. 8.5 - RECYCLING OFF SIT	91. OFF-SITE RECYCLED TOTAL	0.087484811
54. 5.5.1 - LANDFILLS	50. 5.3 - WATER	0.084343174
42. METAL CATEGORY	48. 5.1 - FUGITIVE AIR	0.082739085

112. 8.4 - RECYCLING ON SITE	94. OFF-SITE ENERGY RECOVERY T	0.073973439
42. METAL CATEGORY	114. 8.6 - TREATMENT ON SITE	0.072144664
42. METAL CATEGORY	108. 8.1C - OFF-SITE CONTAIN	0.071930236
48. 5.1 - FUGITIVE AIR	107. 8.1B - ON-SITE OTHER	0.070044813
57. 5.5.2 - LAND TREATMENT	104. TOTAL RELEASES	0.067466420
87. 6.2 - M24	113. 8.5 - RECYCLING OFF SIT	0.066930276
92. 6.2 - M56	112. 8.4 - RECYCLING ON SITE	0.061970701
113. 8.5 - RECYCLING OFF SIT	88. 6.2 - M26	0.061593830
54. 5.5.1 - LANDFILLS	116. PRODUCTION WSTE (8.1-8.7)	0.060849440
113. 8.5 - RECYCLING OFF SIT	42. METAL CATEGORY	0.060789728
50. 5.3 - WATER	48. 5.1 - FUGITIVE AIR	0.060253545
85. OFF-SITE RELEASE TOTAL	42. METAL CATEGORY	0.059240086
111. 8.3 - ENERGY RECOVER OF	42. METAL CATEGORY	0.055233620
116. PRODUCTION WSTE (8.1-8.7)	42. METAL CATEGORY	0.055033046
96. 6.2 - M50	42. METAL CATEGORY	0.054603203
109. 8.1D - OFF-SITE OTHER R	65. POTW - TOTAL TRANSFERS	0.053205673
42. METAL CATEGORY	101. OFF-SITE TREATED TOTAL	0.052980158
42. METAL CATEGORY	92. 6.2 - M56	0.052589042
54. 5.5.1 - LANDFILLS	49. 5.2 - STACK AIR	0.051379880
64. 6.1 - POTW - TRNS TRT	109. 8.1D - OFF-SITE OTHER R	0.049336776
Cancer_Rate_Flag	50. 5.3 - WATER	0.048902138
109. 8.1D - OFF-SITE OTHER R	101. OFF-SITE TREATED TOTAL	0.047811126
42. METAL CATEGORY	115. 8.7 - TREATMENT OFF SITE	0.046083533
50. 5.3 - WATER	42. METAL CATEGORY	0.042797228
63. 6.1 - POTW - TRNS RLSE	85. OFF-SITE RELEASE TOTAL	0.042197423
65. POTW - TOTAL TRANSFERS	42. METAL CATEGORY	0.041660773
57. 5.5.2 - LAND TREATMENT	62. ON-SITE RELEASE TOTAL	0.041587486
60. 5.5.3B - OTHER SURFACE I	42. METAL CATEGORY	0.041557790
112. 8.4 - RECYCLING ON SITE	42. METAL CATEGORY	0.041495935

64. 6.1 - POTW - TRNS TRT	42. METAL CATEGORY	0.041417132
Cancer_Rate_Flag	42. METAL CATEGORY	0.040296882
57. 5.5.2 - LAND TREATMENT	107. 8.1B - ON-SITE OTHER	0.039908894
Cancer_Rate_Flag	116. PRODUCTION WSTE (8.1-8.7)	0.039485779
115. 8.7 - TREATMENT OFF SITE	116. PRODUCTION WSTE (8.1-8.7)	0.038866118
116. PRODUCTION WSTE (8.1-8.7)	101. OFF-SITE TREATED TOTAL	0.038337775
64. 6.1 - POTW - TRNS TRT	116. PRODUCTION WSTE (8.1-8.7)	0.036599421
65. POTW - TOTAL TRANSFERS	116. PRODUCTION WSTE (8.1-8.7)	0.036588385
116. PRODUCTION WSTE (8.1-8.7)	111. 8.3 - ENERGY RECOVER OF	0.033814202
108. 8.1C - OFF-SITE CONTAIN	113. 8.5 - RECYCLING OFF SIT	0.031910161
48. 5.1 - FUGITIVE AIR	96. 6.2 - M50	0.030832615
116. PRODUCTION WSTE (8.1-8.7)	92. 6.2 - M56	0.029679721
113. 8.5 - RECYCLING OFF SIT	85. OFF-SITE RELEASE TOTAL	0.029395647
109. 8.1D - OFF-SITE OTHER R	115. 8.7 - TREATMENT OFF SITE	0.029137241
62. ON-SITE RELEASE TOTAL	54. 5.5.1 - LANDFILLS	0.027153261
85. OFF-SITE RELEASE TOTAL	103. 6.2 - TOTAL TRANSFER	0.026877324
104. TOTAL RELEASES	42. METAL CATEGORY	0.026232331
104. TOTAL RELEASES	54. 5.5.1 - LANDFILLS	0.026147949
48. 5.1 - FUGITIVE AIR	112. 8.4 - RECYCLING ON SITE	0.025753470
42. METAL CATEGORY	57. 5.5.2 - LAND TREATMENT	0.023747425
116. PRODUCTION WSTE (8.1-8.7)	61. 5.5.4 - OTHER DISPOSAL	0.023595706
101. OFF-SITE TREATED TOTAL	103. 6.2 - TOTAL TRANSFER	0.023556763
103. 6.2 - TOTAL TRANSFER	115. 8.7 - TREATMENT OFF SITE	0.023379266
42. METAL CATEGORY	56. 5.5.1B - OTHER LANDFILLS	0.022564962
65. POTW - TOTAL TRANSFERS	103. 6.2 - TOTAL TRANSFER	0.022323476
64. 6.1 - POTW - TRNS TRT	103. 6.2 - TOTAL TRANSFER	0.022306269
109. 8.1D - OFF-SITE OTHER R	116. PRODUCTION WSTE (8.1-8.7)	0.021547540
42. METAL CATEGORY	107. 8.1B - ON-SITE OTHER	0.021071438
69. 6.2 - M40 METAL	104. TOTAL RELEASES	0.019610284

96. 6.2 - M50	101. OFF-SITE TREATED TOTAL	0.019259770
116. PRODUCTION WSTE (8.1-8.7)	108. 8.1C - OFF-SITE CONTAIN	0.018882223
42. METAL CATEGORY	69. 6.2 - M40 METAL	0.018624022
94. OFF-SITE ENERGY RECOVERY T	Cancer_Rate_Flag	0.018442530
107. 8.1B - ON-SITE OTHER	Cancer_Rate_Flag	0.018045120
101. OFF-SITE TREATED TOTAL	Cancer_Rate_Flag	0.017748941
Cancer_Rate_Flag	92. 6.2 - M56	0.017639189
42. METAL CATEGORY	109. 8.1D - OFF-SITE OTHER R	0.017626346
Cancer_Rate_Flag	115. 8.7 - TREATMENT OFF SITE	0.016917033
85. OFF-SITE RELEASE TOTAL	116. PRODUCTION WSTE (8.1-8.7)	0.016414697
42. METAL CATEGORY	63. 6.1 - POTW - TRNS RLSE	0.016304805
50. 5.3 - WATER	108. 8.1C - OFF-SITE CONTAIN	0.015988946
116. PRODUCTION WSTE (8.1-8.7)	103. 6.2 - TOTAL TRANSFER	0.015778242
96. 6.2 - M50	115. 8.7 - TREATMENT OFF SITE	0.014993690
85. OFF-SITE RELEASE TOTAL	61. 5.5.4 - OTHER DISPOSAL	0.014935761
65. POTW - TOTAL TRANSFERS	Cancer_Rate_Flag	0.014270655
64. 6.1 - POTW - TRNS TRT	Cancer_Rate_Flag	0.014254827
91. OFF-SITE RECYCLED TOTAL	116. PRODUCTION WSTE (8.1-8.7)	0.014077119
113. 8.5 - RECYCLING OFF SIT	Cancer_Rate_Flag	0.013569690
Cancer_Rate_Flag	88. 6.2 - M26	0.013565354
112. 8.4 - RECYCLING ON SITE	114. 8.6 - TREATMENT ON SITE	0.013476783
96. 6.2 - M50	54. 5.5.1 - LANDFILLS	0.013235968
42. METAL CATEGORY	62. ON-SITE RELEASE TOTAL	0.012710408
48. 5.1 - FUGITIVE AIR	Cancer_Rate_Flag	0.012656323
103. 6.2 - TOTAL TRANSFER	108. 8.1C - OFF-SITE CONTAIN	0.012343304
91. OFF-SITE RECYCLED TOTAL	42. METAL CATEGORY	0.011983192
109. 8.1D - OFF-SITE OTHER R	108. 8.1C - OFF-SITE CONTAIN	0.011973487
109. 8.1D - OFF-SITE OTHER R	103. 6.2 - TOTAL TRANSFER	0.011820259
103. 6.2 - TOTAL TRANSFER	42. METAL CATEGORY	0.011513610

49. 5.2 - STACK AIR	101. OFF-SITE TREATED TOTAL	0.011197464
96. 6.2 - M50	49. 5.2 - STACK AIR	0.011063358
42. METAL CATEGORY	87. 6.2 - M24	0.011028432
62. ON-SITE RELEASE TOTAL	Cancer_Rate_Flag	0.010894067
94. OFF-SITE ENERGY RECOVERY T	103. 6.2 - TOTAL TRANSFER	0.010731219
49. 5.2 - STACK AIR	115. 8.7 - TREATMENT OFF SITE	0.010706195
109. 8.1D - OFF-SITE OTHER R	112. 8.4 - RECYCLING ON SITE	0.010641982
48. 5.1 - FUGITIVE AIR	94. OFF-SITE ENERGY RECOVERY T	0.010575857
114. 8.6 - TREATMENT ON SITE	Cancer_Rate_Flag	0.010492221
85. OFF-SITE RELEASE TOTAL	115. 8.7 - TREATMENT OFF SITE	0.010450045
116. PRODUCTION WSTE (8.1-8.7)	87. 6.2 - M24	0.010359438
112. 8.4 - RECYCLING ON SITE	96. 6.2 - M50	0.010304861
49. 5.2 - STACK AIR	112. 8.4 - RECYCLING ON SITE	0.010251094
56. 5.5.1B - OTHER LANDFILLS	61. 5.5.4 - OTHER DISPOSAL	0.010237517
92. 6.2 - M56	103. 6.2 - TOTAL TRANSFER	0.009998957
108. 8.1C - OFF-SITE CONTAIN	Cancer_Rate_Flag	0.009856014
65. POTW - TOTAL TRANSFERS	49. 5.2 - STACK AIR	0.009664939
64. 6.1 - POTW - TRNS TRT	49. 5.2 - STACK AIR	0.009580245
56. 5.5.1B - OTHER LANDFILLS	109. 8.1D - OFF-SITE OTHER R	0.009560074
116. PRODUCTION WSTE (8.1-8.7)	63. 6.1 - POTW - TRNS RLSE	0.009516313
96. 6.2 - M50	Cancer_Rate_Flag	0.009434525
114. 8.6 - TREATMENT ON SITE	96. 6.2 - M50	0.009279623
111. 8.3 - ENERGY RECOVER OF	114. 8.6 - TREATMENT ON SITE	0.008883184
49. 5.2 - STACK AIR	88. 6.2 - M26	0.008881562
85. OFF-SITE RELEASE TOTAL	114. 8.6 - TREATMENT ON SITE	0.008616859
Cancer_Rate_Flag	104. TOTAL RELEASES	0.008616771
62. ON-SITE RELEASE TOTAL	101. OFF-SITE TREATED TOTAL	0.008607347
57. 5.5.2 - LAND TREATMENT	Cancer_Rate_Flag	0.008414488
114. 8.6 - TREATMENT ON SITE	92. 6.2 - M56	0.008380275

Cancer_Rate_Flag	85. OFF-SITE RELEASE TOTAL	0.008351335
65. POTW - TOTAL TRANSFERS	48. 5.1 - FUGITIVE AIR	0.008332949
108. 8.1C - OFF-SITE CONTAIN	48. 5.1 - FUGITIVE AIR	0.008316208
111. 8.3 - ENERGY RECOVER OF	50. 5.3 - WATER	0.008295886
64. 6.1 - POTW - TRNS TRT	48. 5.1 - FUGITIVE AIR	0.008267792
114. 8.6 - TREATMENT ON SITE	88. 6.2 - M26	0.008176208
Cancer_Rate_Flag	60. 5.5.3B - OTHER SURFACE I	0.008155970
62. ON-SITE RELEASE TOTAL	115. 8.7 - TREATMENT OFF SITE	0.008150142
108. 8.1C - OFF-SITE CONTAIN	49. 5.2 - STACK AIR	0.008137957
109. 8.1D - OFF-SITE OTHER R	49. 5.2 - STACK AIR	0.008133978
48. 5.1 - FUGITIVE AIR	88. 6.2 - M26	0.008133851
101. OFF-SITE TREATED TOTAL	85. OFF-SITE RELEASE TOTAL	0.008089588
113. 8.5 - RECYCLING OFF SIT	114. 8.6 - TREATMENT ON SITE	0.008017966
48. 5.1 - FUGITIVE AIR	101. OFF-SITE TREATED TOTAL	0.007951045
92. 6.2 - M56	50. 5.3 - WATER	0.007908826
96. 6.2 - M50	116. PRODUCTION WSTE (8.1-8.7)	0.007870601
85. OFF-SITE RELEASE TOTAL	65. POTW - TOTAL TRANSFERS	0.007827384
112. 8.4 - RECYCLING ON SITE	50. 5.3 - WATER	0.007541552
101. OFF-SITE TREATED TOTAL	50. 5.3 - WATER	0.007463739
114. 8.6 - TREATMENT ON SITE	101. OFF-SITE TREATED TOTAL	0.007391908
114. 8.6 - TREATMENT ON SITE	115. 8.7 - TREATMENT OFF SITE	0.007319753
65. POTW - TOTAL TRANSFERS	62. ON-SITE RELEASE TOTAL	0.007227801
64. 6.1 - POTW - TRNS TRT	62. ON-SITE RELEASE TOTAL	0.007167411
115. 8.7 - TREATMENT OFF SITE	48. 5.1 - FUGITIVE AIR	0.007124662
115. 8.7 - TREATMENT OFF SITE	50. 5.3 - WATER	0.007112640
114. 8.6 - TREATMENT ON SITE	65. POTW - TOTAL TRANSFERS	0.007078426
64. 6.1 - POTW - TRNS TRT	114. 8.6 - TREATMENT ON SITE	0.007022166
113. 8.5 - RECYCLING OFF SIT	49. 5.2 - STACK AIR	0.006954146
48. 5.1 - FUGITIVE AIR	113. 8.5 - RECYCLING OFF SIT	0.006931494

114. 8.6 - TREATMENT ON SITE	108. 8.1C - OFF-SITE CONTAIN	0.006881176
88. 6.2 - M26	104. TOTAL RELEASES	0.006790273
104. TOTAL RELEASES	63. 6.1 - POTW - TRNS RLSE	0.006775138
62. ON-SITE RELEASE TOTAL	88. 6.2 - M26	0.006762916
101. OFF-SITE TREATED TOTAL	111. 8.3 - ENERGY RECOVER OF	0.006656488
42. METAL CATEGORY	119. 8.9 - PRODUCTION RATIO	0.006625184
88. 6.2 - M26	42. METAL CATEGORY	0.006594113
61. 5.5.4 - OTHER DISPOSAL	Cancer_Rate_Flag	0.006564231
109. 8.1D - OFF-SITE OTHER R	114. 8.6 - TREATMENT ON SITE	0.006516802
104. TOTAL RELEASES	101. OFF-SITE TREATED TOTAL	0.006458323
94. OFF-SITE ENERGY RECOVERY T	115. 8.7 - TREATMENT OFF SITE	0.006324993
63. 6.1 - POTW - TRNS RLSE	103. 6.2 - TOTAL TRANSFER	0.006317192
92. 6.2 - M56	101. OFF-SITE TREATED TOTAL	0.006278825
113. 8.5 - RECYCLING OFF SIT	50. 5.3 - WATER	0.006221889
50. 5.3 - WATER	65. POTW - TOTAL TRANSFERS	0.006085674
92. 6.2 - M56	104. TOTAL RELEASES	0.006052332
50. 5.3 - WATER	64. 6.1 - POTW - TRNS TRT	0.006037616
54. 5.5.1 - LANDFILLS	48. 5.1 - FUGITIVE AIR	0.006009291
85. OFF-SITE RELEASE TOTAL	50. 5.3 - WATER	0.005995149
96. 6.2 - M50	88. 6.2 - M26	0.005979725
85. OFF-SITE RELEASE TOTAL	64. 6.1 - POTW - TRNS TRT	0.005948360
92. 6.2 - M56	115. 8.7 - TREATMENT OFF SITE	0.005901755
49. 5.2 - STACK AIR	61. 5.5.4 - OTHER DISPOSAL	0.005880039
85. OFF-SITE RELEASE TOTAL	62. ON-SITE RELEASE TOTAL	0.005792130
48. 5.1 - FUGITIVE AIR	92. 6.2 - M56	0.005718467
94. OFF-SITE ENERGY RECOVERY T	108. 8.1C - OFF-SITE CONTAIN	0.005676003
104. TOTAL RELEASES	111. 8.3 - ENERGY RECOVER OF	0.005673043
94. OFF-SITE ENERGY RECOVERY T	65. POTW - TOTAL TRANSFERS	0.005592362
88. 6.2 - M26	111. 8.3 - ENERGY RECOVER OF	0.005589426

57. 5.5.2 - LAND TREATMENT	49. 5.2 - STACK AIR	0.005568406
104. TOTAL RELEASES	64. 6.1 - POTW - TRNS TRT	0.005562691
94. OFF-SITE ENERGY RECOVERY T	64. 6.1 - POTW - TRNS TRT	0.005543258
48. 5.1 - FUGITIVE AIR	61. 5.5.4 - OTHER DISPOSAL	0.005525155
107. 8.1B - ON-SITE OTHER	101. OFF-SITE TREATED TOTAL	0.005524261
115. 8.7 - TREATMENT OFF SITE	104. TOTAL RELEASES	0.005460889
92. 6.2 - M56	108. 8.1C - OFF-SITE CONTAIN	0.005407836
85. OFF-SITE RELEASE TOTAL	49. 5.2 - STACK AIR	0.005364771
101. OFF-SITE TREATED TOTAL	113. 8.5 - RECYCLING OFF SIT	0.005358353
65. POTW - TOTAL TRANSFERS	92. 6.2 - M56	0.005326974
92. 6.2 - M56	88. 6.2 - M26	0.005297719
62. ON-SITE RELEASE TOTAL	92. 6.2 - M56	0.005281212
92. 6.2 - M56	64. 6.1 - POTW - TRNS TRT	0.005280141
115. 8.7 - TREATMENT OFF SITE	107. 8.1B - ON-SITE OTHER	0.005267987
104. TOTAL RELEASES	65. POTW - TOTAL TRANSFERS	0.005180623
61. 5.5.4 - OTHER DISPOSAL	109. 8.1D - OFF-SITE OTHER R	0.005163031
115. 8.7 - TREATMENT OFF SITE	113. 8.5 - RECYCLING OFF SIT	0.005089226
103. 6.2 - TOTAL TRANSFER	104. TOTAL RELEASES	0.005069452
113. 8.5 - RECYCLING OFF SIT	62. ON-SITE RELEASE TOTAL	0.005030826
92. 6.2 - M56	107. 8.1B - ON-SITE OTHER	0.005006927
91. OFF-SITE RECYCLED TOTAL	88. 6.2 - M26	0.004992047
112. 8.4 - RECYCLING ON SITE	65. POTW - TOTAL TRANSFERS	0.004982791
61. 5.5.4 - OTHER DISPOSAL	114. 8.6 - TREATMENT ON SITE	0.004967309
112. 8.4 - RECYCLING ON SITE	64. 6.1 - POTW - TRNS TRT	0.004966654
111. 8.3 - ENERGY RECOVER OF	62. ON-SITE RELEASE TOTAL	0.004944626
116. PRODUCTION WSTE (8.1-8.7)	57. 5.5.2 - LAND TREATMENT	0.004934765
96. 6.2 - M50	109. 8.1D - OFF-SITE OTHER R	0.004905686
111. 8.3 - ENERGY RECOVER OF	107. 8.1B - ON-SITE OTHER	0.004846696
88. 6.2 - M26	103. 6.2 - TOTAL TRANSFER	0.004846514

109. 8.1D - OFF-SITE OTHER R	50. 5.3 - WATER	0.004756353
96. 6.2 - M50	65. POTW - TOTAL TRANSFERS	0.004748785
115. 8.7 - TREATMENT OFF SITE	88. 6.2 - M26	0.004742615
113. 8.5 - RECYCLING OFF SIT	111. 8.3 - ENERGY RECOVER OF	0.004741343
114. 8.6 - TREATMENT ON SITE	57. 5.5.2 - LAND TREATMENT	0.004701430
96. 6.2 - M50	64. 6.1 - POTW - TRNS TRT	0.004699697
112. 8.4 - RECYCLING ON SITE	107. 8.1B - ON-SITE OTHER	0.004663017
113. 8.5 - RECYCLING OFF SIT	112. 8.4 - RECYCLING ON SITE	0.004557170
50. 5.3 - WATER	88. 6.2 - M26	0.004537242
65. POTW - TOTAL TRANSFERS	107. 8.1B - ON-SITE OTHER	0.004526404
101. OFF-SITE TREATED TOTAL	112. 8.4 - RECYCLING ON SITE	0.004517979
107. 8.1B - ON-SITE OTHER	88. 6.2 - M26	0.004511310
107. 8.1B - ON-SITE OTHER	64. 6.1 - POTW - TRNS TRT	0.004490833
49. 5.2 - STACK AIR	63. 6.1 - POTW - TRNS RLSE	0.004383935
108. 8.1C - OFF-SITE CONTAIN	65. POTW - TOTAL TRANSFERS	0.004348597
108. 8.1C - OFF-SITE CONTAIN	64. 6.1 - POTW - TRNS TRT	0.004315211
119. 8.9 - PRODUCTION RATIO	109. 8.1D - OFF-SITE OTHER R	0.004313610
108. 8.1C - OFF-SITE CONTAIN	96. 6.2 - M50	0.004292459
92. 6.2 - M56	49. 5.2 - STACK AIR	0.004278218
113. 8.5 - RECYCLING OFF SIT	65. POTW - TOTAL TRANSFERS	0.004242120
113. 8.5 - RECYCLING OFF SIT	64. 6.1 - POTW - TRNS TRT	0.004219872
115. 8.7 - TREATMENT OFF SITE	112. 8.4 - RECYCLING ON SITE	0.004190343
Cancer_Rate_Flag	112. 8.4 - RECYCLING ON SITE	0.004188708
49. 5.2 - STACK AIR	60. 5.5.3B - OTHER SURFACE I	0.004160465
113. 8.5 - RECYCLING OFF SIT	107. 8.1B - ON-SITE OTHER	0.004148029
Cancer_Rate_Flag	63. 6.1 - POTW - TRNS RLSE	0.004141446
113. 8.5 - RECYCLING OFF SIT	109. 8.1D - OFF-SITE OTHER R	0.004003866
85. OFF-SITE RELEASE TOTAL	92. 6.2 - M56	0.003946545
69. 6.2 - M40 METAL	Cancer_Rate_Flag	0.003935659

49. 5.2 - STACK AIR	94. OFF-SITE ENERGY RECOVERY T	0.003775463
49. 5.2 - STACK AIR	56. 5.5.1B - OTHER LANDFILLS	0.003726862
85. OFF-SITE RELEASE TOTAL	111. 8.3 - ENERGY RECOVER OF	0.003722490
61. 5.5.4 - OTHER DISPOSAL	94. OFF-SITE ENERGY RECOVERY T	0.003660437
96. 6.2 - M50	61. 5.5.4 - OTHER DISPOSAL	0.003642600
85. OFF-SITE RELEASE TOTAL	48. 5.1 - FUGITIVE AIR	0.003641282
48. 5.1 - FUGITIVE AIR	63. 6.1 - POTW - TRNS RLSE	0.003611862
60. 5.5.3B - OTHER SURFACE I	114. 8.6 - TREATMENT ON SITE	0.003599772
96. 6.2 - M50	57. 5.5.2 - LAND TREATMENT	0.003487286
92. 6.2 - M56	61. 5.5.4 - OTHER DISPOSAL	0.003487220
101. OFF-SITE TREATED TOTAL	61. 5.5.4 - OTHER DISPOSAL	0.003484290
111. 8.3 - ENERGY RECOVER OF	57. 5.5.2 - LAND TREATMENT	0.003481740
103. 6.2 - TOTAL TRANSFER	Cancer_Rate_Flag	0.003434131
101. OFF-SITE TREATED TOTAL	57. 5.5.2 - LAND TREATMENT	0.003359119
61. 5.5.4 - OTHER DISPOSAL	115. 8.7 - TREATMENT OFF SITE	0.003330883
92. 6.2 - M56	57. 5.5.2 - LAND TREATMENT	0.003319805
61. 5.5.4 - OTHER DISPOSAL	112. 8.4 - RECYCLING ON SITE	0.003245496
63. 6.1 - POTW - TRNS RLSE	62. ON-SITE RELEASE TOTAL	0.003215555
69. 6.2 - M40 METAL	115. 8.7 - TREATMENT OFF SITE	0.003178803
85. OFF-SITE RELEASE TOTAL	56. 5.5.1B - OTHER LANDFILLS	0.003148560
114. 8.6 - TREATMENT ON SITE	63. 6.1 - POTW - TRNS RLSE	0.003087567
61. 5.5.4 - OTHER DISPOSAL	50. 5.3 - WATER	0.003086907
107. 8.1B - ON-SITE OTHER	96. 6.2 - M50	0.003079650
94. OFF-SITE ENERGY RECOVERY T	109. 8.1D - OFF-SITE OTHER R	0.002996414
109. 8.1D - OFF-SITE OTHER R	60. 5.5.3B - OTHER SURFACE I	0.002987198
85. OFF-SITE RELEASE TOTAL	112. 8.4 - RECYCLING ON SITE	0.002984388
42. METAL CATEGORY	54. 5.5.1 - LANDFILLS	0.002920233
61. 5.5.4 - OTHER DISPOSAL	88. 6.2 - M26	0.002864330
112. 8.4 - RECYCLING ON SITE	57. 5.5.2 - LAND TREATMENT	0.002860573

48. 5.1 - FUGITIVE AIR	60. 5.5.3B - OTHER SURFACE I	0.002812362
104. TOTAL RELEASES	96. 6.2 - M50	0.002765698
61. 5.5.4 - OTHER DISPOSAL	65. POTW - TOTAL TRANSFERS	0.002757858
61. 5.5.4 - OTHER DISPOSAL	64. 6.1 - POTW - TRNS TRT	0.002735776
69. 6.2 - M40 METAL	49. 5.2 - STACK AIR	0.002722873
92. 6.2 - M56	109. 8.1D - OFF-SITE OTHER R	0.002715991
57. 5.5.2 - LAND TREATMENT	48. 5.1 - FUGITIVE AIR	0.002688440
96. 6.2 - M50	60. 5.5.3B - OTHER SURFACE I	0.002683838
60. 5.5.3B - OTHER SURFACE I	94. OFF-SITE ENERGY RECOVERY T	0.002649988
63. 6.1 - POTW - TRNS RLSE	50. 5.3 - WATER	0.002647911
57. 5.5.2 - LAND TREATMENT	65. POTW - TOTAL TRANSFERS	0.002631129
54. 5.5.1 - LANDFILLS	Cancer_Rate_Flag	0.002627724
61. 5.5.4 - OTHER DISPOSAL	57. 5.5.2 - LAND TREATMENT	0.002623752
112. 8.4 - RECYCLING ON SITE	88. 6.2 - M26	0.002617623
57. 5.5.2 - LAND TREATMENT	64. 6.1 - POTW - TRNS TRT	0.002609701
56. 5.5.1B - OTHER LANDFILLS	Cancer_Rate_Flag	0.002589857
Cancer_Rate_Flag	91. OFF-SITE RECYCLED TOTAL	0.002585025
Cancer_Rate_Flag	109. 8.1D - OFF-SITE OTHER R	0.002566898
60. 5.5.3B - OTHER SURFACE I	101. OFF-SITE TREATED TOTAL	0.002557278
113. 8.5 - RECYCLING OFF SIT	61. 5.5.4 - OTHER DISPOSAL	0.002545797
63. 6.1 - POTW - TRNS RLSE	111. 8.3 - ENERGY RECOVER OF	0.002538744
113. 8.5 - RECYCLING OFF SIT	57. 5.5.2 - LAND TREATMENT	0.002538628
92. 6.2 - M56	60. 5.5.3B - OTHER SURFACE I	0.002527026
56. 5.5.1B - OTHER LANDFILLS	94. OFF-SITE ENERGY RECOVERY T	0.002460431
56. 5.5.1B - OTHER LANDFILLS	114. 8.6 - TREATMENT ON SITE	0.002454848
115. 8.7 - TREATMENT OFF SITE	60. 5.5.3B - OTHER SURFACE I	0.002449633
96. 6.2 - M50	56. 5.5.1B - OTHER LANDFILLS	0.002435787
48. 5.1 - FUGITIVE AIR	69. 6.2 - M40 METAL	0.002426744
91. OFF-SITE RECYCLED TOTAL	108. 8.1C - OFF-SITE CONTAIN	0.002423744

63. 6.1 - POTW - TRNS RLSE	92. 6.2 - M56	0.002419536
116. PRODUCTION WSTE (8.1-8.7)	88. 6.2 - M26	0.002389353
69. 6.2 - M40 METAL	116. PRODUCTION WSTE (8.1-8.7)	0.002386822
56. 5.5.1B - OTHER LANDFILLS	101. OFF-SITE TREATED TOTAL	0.002372884
112. 8.4 - RECYCLING ON SITE	60. 5.5.3B - OTHER SURFACE I	0.002352022
92. 6.2 - M56	56. 5.5.1B - OTHER LANDFILLS	0.002346219
62. ON-SITE RELEASE TOTAL	96. 6.2 - M50	0.002314124
63. 6.1 - POTW - TRNS RLSE	96. 6.2 - M50	0.002313561
56. 5.5.1B - OTHER LANDFILLS	115. 8.7 - TREATMENT OFF SITE	0.002272696
92. 6.2 - M56	96. 6.2 - M50	0.002251215
103. 6.2 - TOTAL TRANSFER	57. 5.5.2 - LAND TREATMENT	0.002248810
112. 8.4 - RECYCLING ON SITE	56. 5.5.1B - OTHER LANDFILLS	0.002247034
56. 5.5.1B - OTHER LANDFILLS	113. 8.5 - RECYCLING OFF SIT	0.002240019
Cancer_Rate_Flag	49. 5.2 - STACK AIR	0.002235158
87. 6.2 - M24	Cancer_Rate_Flag	0.002231484
88. 6.2 - M26	101. OFF-SITE TREATED TOTAL	0.002219925
96. 6.2 - M50	85. OFF-SITE RELEASE TOTAL	0.002214025
60. 5.5.3B - OTHER SURFACE I	50. 5.3 - WATER	0.002213877
61. 5.5.4 - OTHER DISPOSAL	108. 8.1C - OFF-SITE CONTAIN	0.002176598
Cancer_Rate_Flag	119. 8.9 - PRODUCTION RATIO	0.002127026
109. 8.1D - OFF-SITE OTHER R	62. ON-SITE RELEASE TOTAL	0.002126102
91. OFF-SITE RECYCLED TOTAL	85. OFF-SITE RELEASE TOTAL	0.002115809
69. 6.2 - M40 METAL	114. 8.6 - TREATMENT ON SITE	0.002100549
60. 5.5.3B - OTHER SURFACE I	88. 6.2 - M26	0.002082531
96. 6.2 - M50	113. 8.5 - RECYCLING OFF SIT	0.002059074
103. 6.2 - TOTAL TRANSFER	49. 5.2 - STACK AIR	0.002020719
113. 8.5 - RECYCLING OFF SIT	104. TOTAL RELEASES	0.002010538
65. POTW - TOTAL TRANSFERS	60. 5.5.3B - OTHER SURFACE I	0.002000903
113. 8.5 - RECYCLING OFF SIT	60. 5.5.3B - OTHER SURFACE I	0.001998644

62. ON-SITE RELEASE TOTAL	69. 6.2 - M40 METAL	0.001992834
69. 6.2 - M40 METAL	103. 6.2 - TOTAL TRANSFER	0.001990641
87. 6.2 - M24	85. OFF-SITE RELEASE TOTAL	0.001988111
64. 6.1 - POTW - TRNS TRT	60. 5.5.3B - OTHER SURFACE I	0.001984610
108. 8.1C - OFF-SITE CONTAIN	60. 5.5.3B - OTHER SURFACE I	0.001983217
112. 8.4 - RECYCLING ON SITE	104. TOTAL RELEASES	0.001966653
63. 6.1 - POTW - TRNS RLSE	107. 8.1B - ON-SITE OTHER	0.001965764
114. 8.6 - TREATMENT ON SITE	103. 6.2 - TOTAL TRANSFER	0.001942751
87. 6.2 - M24	108. 8.1C - OFF-SITE CONTAIN	0.001942141
57. 5.5.2 - LAND TREATMENT	88. 6.2 - M26	0.001940728
112. 8.4 - RECYCLING ON SITE	108. 8.1C - OFF-SITE CONTAIN	0.001879271
63. 6.1 - POTW - TRNS RLSE	108. 8.1C - OFF-SITE CONTAIN	0.001871706
56. 5.5.1B - OTHER LANDFILLS	65. POTW - TOTAL TRANSFERS	0.001857823
56. 5.5.1B - OTHER LANDFILLS	64. 6.1 - POTW - TRNS TRT	0.001842730
63. 6.1 - POTW - TRNS RLSE	88. 6.2 - M26	0.001833847
109. 8.1D - OFF-SITE OTHER R	107. 8.1B - ON-SITE OTHER	0.001824771
63. 6.1 - POTW - TRNS RLSE	112. 8.4 - RECYCLING ON SITE	0.001672538
69. 6.2 - M40 METAL	50. 5.3 - WATER	0.001670279
103. 6.2 - TOTAL TRANSFER	48. 5.1 - FUGITIVE AIR	0.001635751
63. 6.1 - POTW - TRNS RLSE	113. 8.5 - RECYCLING OFF SIT	0.001605569
119. 8.9 - PRODUCTION RATIO	85. OFF-SITE RELEASE TOTAL	0.001586097
49. 5.2 - STACK AIR	91. OFF-SITE RECYCLED TOTAL	0.001572772
96. 6.2 - M50	69. 6.2 - M40 METAL	0.001566658
56. 5.5.1B - OTHER LANDFILLS	88. 6.2 - M26	0.001566617
69. 6.2 - M40 METAL	94. OFF-SITE ENERGY RECOVERY T	0.001546809
48. 5.1 - FUGITIVE AIR	109. 8.1D - OFF-SITE OTHER R	0.001511238
69. 6.2 - M40 METAL	101. OFF-SITE TREATED TOTAL	0.001492696
92. 6.2 - M56	69. 6.2 - M40 METAL	0.001475022
62. ON-SITE RELEASE TOTAL	108. 8.1C - OFF-SITE CONTAIN	0.001474367

48. 5.1 - FUGITIVE AIR	91. OFF-SITE RECYCLED TOTAL	0.001457580
91. OFF-SITE RECYCLED TOTAL	114. 8.6 - TREATMENT ON SITE	0.001448867
69. 6.2 - M40 METAL	112. 8.4 - RECYCLING ON SITE	0.001409196
87. 6.2 - M24	48. 5.1 - FUGITIVE AIR	0.001401520
49. 5.2 - STACK AIR	87. 6.2 - M24	0.001366965
112. 8.4 - RECYCLING ON SITE	62. ON-SITE RELEASE TOTAL	0.001304885
119. 8.9 - PRODUCTION RATIO	50. 5.3 - WATER	0.001289879
62. ON-SITE RELEASE TOTAL	103. 6.2 - TOTAL TRANSFER	0.001271323
50. 5.3 - WATER	103. 6.2 - TOTAL TRANSFER	0.001267990
69. 6.2 - M40 METAL	107. 8.1B - ON-SITE OTHER	0.001245649
87. 6.2 - M24	114. 8.6 - TREATMENT ON SITE	0.001225449
50. 5.3 - WATER	96. 6.2 - M50	0.001220304
69. 6.2 - M40 METAL	88. 6.2 - M26	0.001218382
61. 5.5.4 - OTHER DISPOSAL	63. 6.1 - POTW - TRNS RLSE	0.001206431
85. OFF-SITE RELEASE TOTAL	107. 8.1B - ON-SITE OTHER	0.001203581
69. 6.2 - M40 METAL	113. 8.5 - RECYCLING OFF SIT	0.001190939
57. 5.5.2 - LAND TREATMENT	60. 5.5.3B - OTHER SURFACE I	0.001189925
49. 5.2 - STACK AIR	119. 8.9 - PRODUCTION RATIO	0.001174314
65. POTW - TOTAL TRANSFERS	69. 6.2 - M40 METAL	0.001167929
108. 8.1C - OFF-SITE CONTAIN	69. 6.2 - M40 METAL	0.001159016
57. 5.5.2 - LAND TREATMENT	63. 6.1 - POTW - TRNS RLSE	0.001158684
64. 6.1 - POTW - TRNS TRT	69. 6.2 - M40 METAL	0.001158425
65. POTW - TOTAL TRANSFERS	88. 6.2 - M26	0.001151925
56. 5.5.1B - OTHER LANDFILLS	108. 8.1C - OFF-SITE CONTAIN	0.001150124
91. OFF-SITE RECYCLED TOTAL	62. ON-SITE RELEASE TOTAL	0.001146211
50. 5.3 - WATER	91. OFF-SITE RECYCLED TOTAL	0.001138770
64. 6.1 - POTW - TRNS TRT	88. 6.2 - M26	0.001080400
48. 5.1 - FUGITIVE AIR	119. 8.9 - PRODUCTION RATIO	0.001061695
116. PRODUCTION WSTE (8.1-8.7)	119. 8.9 - PRODUCTION RATIO	0.001047473

57. 5.5.2 - LAND TREATMENT	56. 5.5.1B - OTHER LANDFILLS	0.001045330
94. OFF-SITE ENERGY RECOVERY T	54. 5.5.1 - LANDFILLS	0.001032759
103. 6.2 - TOTAL TRANSFER	107. 8.1B - ON-SITE OTHER	0.001025227
91. OFF-SITE RECYCLED TOTAL	101. OFF-SITE TREATED TOTAL	0.000999246
109. 8.1D - OFF-SITE OTHER R	69. 6.2 - M40 METAL	0.000988693
92. 6.2 - M56	54. 5.5.1 - LANDFILLS	0.000984838
87. 6.2 - M24	62. ON-SITE RELEASE TOTAL	0.000980185
50. 5.3 - WATER	87. 6.2 - M24	0.000970837
85. OFF-SITE RELEASE TOTAL	88. 6.2 - M26	0.000960067
115. 8.7 - TREATMENT OFF SITE	91. OFF-SITE RECYCLED TOTAL	0.000922726
87. 6.2 - M24	111. 8.3 - ENERGY RECOVER OF	0.000902827
54. 5.5.1 - LANDFILLS	85. OFF-SITE RELEASE TOTAL	0.000900110
87. 6.2 - M24	96. 6.2 - M50	0.000882953
63. 6.1 - POTW - TRNS RLSE	60. 5.5.3B - OTHER SURFACE I	0.000881105
87. 6.2 - M24	101. OFF-SITE TREATED TOTAL	0.000869372
119. 8.9 - PRODUCTION RATIO	114. 8.6 - TREATMENT ON SITE	0.000867864
113. 8.5 - RECYCLING OFF SIT	92. 6.2 - M56	0.000864474
87. 6.2 - M24	92. 6.2 - M56	0.000860790
119. 8.9 - PRODUCTION RATIO	101. OFF-SITE TREATED TOTAL	0.000849337
111. 8.3 - ENERGY RECOVER OF	119. 8.9 - PRODUCTION RATIO	0.000840314
54. 5.5.1 - LANDFILLS	107. 8.1B - ON-SITE OTHER	0.000831683
119. 8.9 - PRODUCTION RATIO	96. 6.2 - M50	0.000822533
56. 5.5.1B - OTHER LANDFILLS	60. 5.5.3B - OTHER SURFACE I	0.000821432
63. 6.1 - POTW - TRNS RLSE	56. 5.5.1B - OTHER LANDFILLS	0.000817365
88. 6.2 - M26	109. 8.1D - OFF-SITE OTHER R	0.000815173
88. 6.2 - M26	54. 5.5.1 - LANDFILLS	0.000813478
112. 8.4 - RECYCLING ON SITE	119. 8.9 - PRODUCTION RATIO	0.000812605
115. 8.7 - TREATMENT OFF SITE	119. 8.9 - PRODUCTION RATIO	0.000809750
107. 8.1B - ON-SITE OTHER	91. OFF-SITE RECYCLED TOTAL	0.000805115

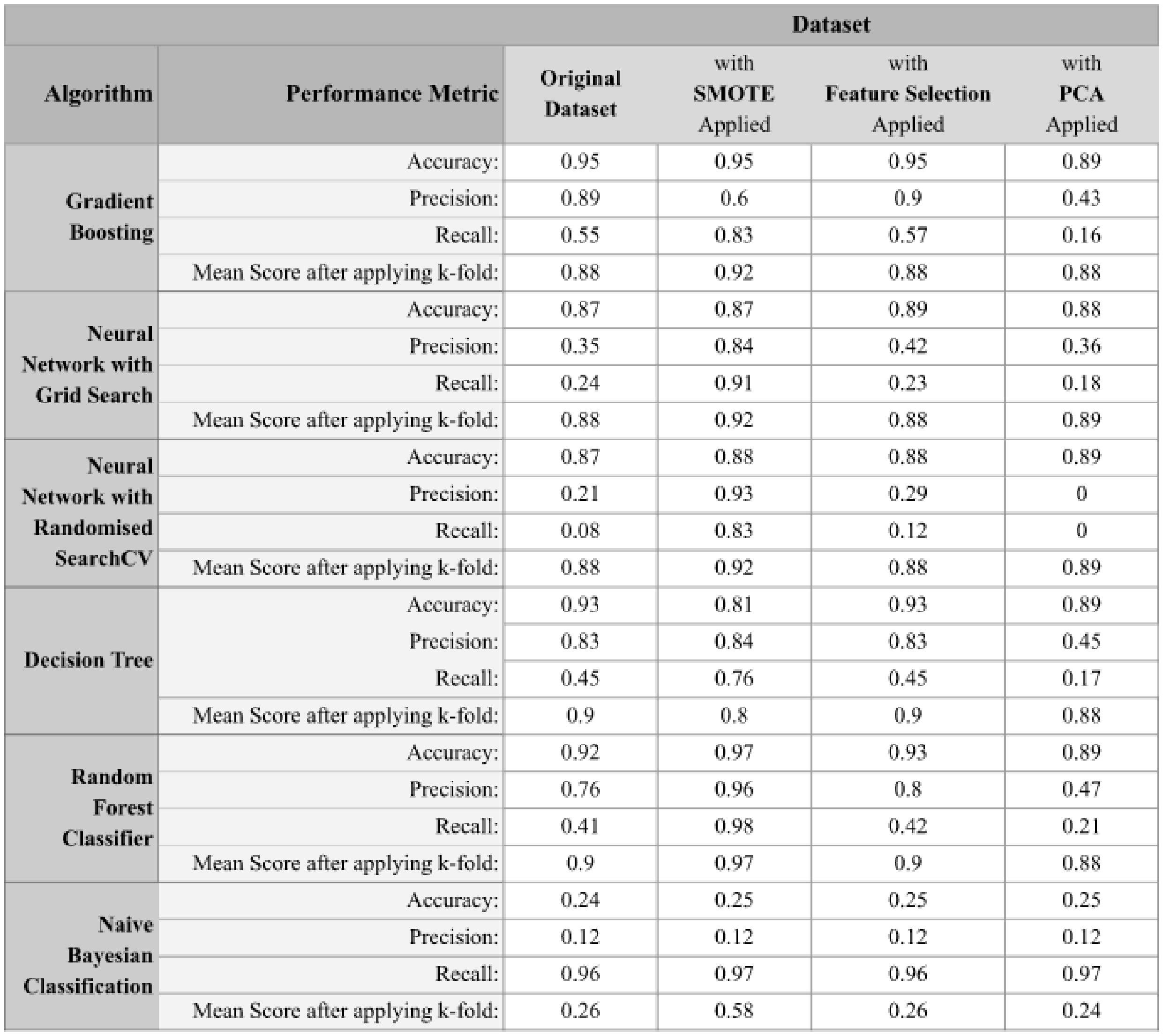
119. 8.9 - PRODUCTION RATIO	92. 6.2 - M56	0.000796752
112. 8.4 - RECYCLING ON SITE	87. 6.2 - M24	0.000790911
115. 8.7 - TREATMENT OFF SITE	87. 6.2 - M24	0.000787130
91. OFF-SITE RECYCLED TOTAL	65. POTW - TOTAL TRANSFERS	0.000786310
54. 5.5.1 - LANDFILLS	113. 8.5 - RECYCLING OFF SIT	0.000782234
64. 6.1 - POTW - TRNS TRT	91. OFF-SITE RECYCLED TOTAL	0.000780906
108. 8.1C - OFF-SITE CONTAIN	54. 5.5.1 - LANDFILLS	0.000773841
54. 5.5.1 - LANDFILLS	64. 6.1 - POTW - TRNS TRT	0.000773446
115. 8.7 - TREATMENT OFF SITE	57. 5.5.2 - LAND TREATMENT	0.000772913
54. 5.5.1 - LANDFILLS	65. POTW - TOTAL TRANSFERS	0.000772427
69. 6.2 - M40 METAL	61. 5.5.4 - OTHER DISPOSAL	0.000736378
91. OFF-SITE RECYCLED TOTAL	96. 6.2 - M50	0.000732072
54. 5.5.1 - LANDFILLS	112. 8.4 - RECYCLING ON SITE	0.000710236
107. 8.1B - ON-SITE OTHER	87. 6.2 - M24	0.000704283
57. 5.5.2 - LAND TREATMENT	69. 6.2 - M40 METAL	0.000700740
112. 8.4 - RECYCLING ON SITE	103. 6.2 - TOTAL TRANSFER	0.000687056
87. 6.2 - M24	65. POTW - TOTAL TRANSFERS	0.000680362
64. 6.1 - POTW - TRNS TRT	87. 6.2 - M24	0.000676045
119. 8.9 - PRODUCTION RATIO	65. POTW - TOTAL TRANSFERS	0.000663405
109. 8.1D - OFF-SITE OTHER R	54. 5.5.1 - LANDFILLS	0.000660121
119. 8.9 - PRODUCTION RATIO	64. 6.1 - POTW - TRNS TRT	0.000657728
101. OFF-SITE TREATED TOTAL	54. 5.5.1 - LANDFILLS	0.000642178
104. TOTAL RELEASES	91. OFF-SITE RECYCLED TOTAL	0.000616461
92. 6.2 - M56	91. OFF-SITE RECYCLED TOTAL	0.000608084
116. PRODUCTION WSTE (8.1-8.7)	56. 5.5.1B - OTHER LANDFILLS	0.000600651
54. 5.5.1 - LANDFILLS	115. 8.7 - TREATMENT OFF SITE	0.000595467
108. 8.1C - OFF-SITE CONTAIN	119. 8.9 - PRODUCTION RATIO	0.000581052
94. OFF-SITE ENERGY RECOVERY T	96. 6.2 - M50	0.000580433
107. 8.1B - ON-SITE OTHER	108. 8.1C - OFF-SITE CONTAIN	0.000576648

69. 6.2 - M40 METAL	60. 5.5.3B - OTHER SURFACE I	0.000532868
56. 5.5.1B - OTHER LANDFILLS	50. 5.3 - WATER	0.000527291
69. 6.2 - M40 METAL	63. 6.1 - POTW - TRNS RLSE	0.000514180
88. 6.2 - M26	87. 6.2 - M24	0.000503124
119. 8.9 - PRODUCTION RATIO	88. 6.2 - M26	0.000502327
56. 5.5.1B - OTHER LANDFILLS	69. 6.2 - M40 METAL	0.000494773
61. 5.5.4 - OTHER DISPOSAL	54. 5.5.1 - LANDFILLS	0.000491658
104. TOTAL RELEASES	87. 6.2 - M24	0.000485240
61. 5.5.4 - OTHER DISPOSAL	91. OFF-SITE RECYCLED TOTAL	0.000484385
119. 8.9 - PRODUCTION RATIO	113. 8.5 - RECYCLING OFF SIT	0.000475238
91. OFF-SITE RECYCLED TOTAL	57. 5.5.2 - LAND TREATMENT	0.000471132
54. 5.5.1 - LANDFILLS	57. 5.5.2 - LAND TREATMENT	0.000467864
115. 8.7 - TREATMENT OFF SITE	108. 8.1C - OFF-SITE CONTAIN	0.000462904
103. 6.2 - TOTAL TRANSFER	60. 5.5.3B - OTHER SURFACE I	0.000462440
107. 8.1B - ON-SITE OTHER	56. 5.5.1B - OTHER LANDFILLS	0.000460890
61. 5.5.4 - OTHER DISPOSAL	119. 8.9 - PRODUCTION RATIO	0.000430165
87. 6.2 - M24	61. 5.5.4 - OTHER DISPOSAL	0.000414811
87. 6.2 - M24	57. 5.5.2 - LAND TREATMENT	0.000403678
57. 5.5.2 - LAND TREATMENT	119. 8.9 - PRODUCTION RATIO	0.000374612
109. 8.1D - OFF-SITE OTHER R	57. 5.5.2 - LAND TREATMENT	0.000374175
91. OFF-SITE RECYCLED TOTAL	60. 5.5.3B - OTHER SURFACE I	0.000364313
54. 5.5.1 - LANDFILLS	60. 5.5.3B - OTHER SURFACE I	0.000355780
54. 5.5.1 - LANDFILLS	56. 5.5.1B - OTHER LANDFILLS	0.000330346
91. OFF-SITE RECYCLED TOTAL	63. 6.1 - POTW - TRNS RLSE	0.000324925
108. 8.1C - OFF-SITE CONTAIN	88. 6.2 - M26	0.000314880
60. 5.5.3B - OTHER SURFACE I	87. 6.2 - M24	0.000308431
119. 8.9 - PRODUCTION RATIO	63. 6.1 - POTW - TRNS RLSE	0.000298008
119. 8.9 - PRODUCTION RATIO	62. ON-SITE RELEASE TOTAL	0.000275356
87. 6.2 - M24	63. 6.1 - POTW - TRNS RLSE	0.000273482

119. 8.9 - PRODUCTION RATIO	56. 5.5.1B - OTHER LANDFILLS	0.000262111
61. 5.5.4 - OTHER DISPOSAL	103. 6.2 - TOTAL TRANSFER	0.000243747
48. 5.1 - FUGITIVE AIR	56. 5.5.1B - OTHER LANDFILLS	0.000224637
85. OFF-SITE RELEASE TOTAL	60. 5.5.3B - OTHER SURFACE I	0.000224175
60. 5.5.3B - OTHER SURFACE I	119. 8.9 - PRODUCTION RATIO	0.000217418
69. 6.2 - M40 METAL	91. OFF-SITE RECYCLED TOTAL	0.000214804
54. 5.5.1 - LANDFILLS	69. 6.2 - M40 METAL	0.000207670
54. 5.5.1 - LANDFILLS	103. 6.2 - TOTAL TRANSFER	0.000191848
96. 6.2 - M50	103. 6.2 - TOTAL TRANSFER	0.000187459
63. 6.1 - POTW - TRNS RLSE	54. 5.5.1 - LANDFILLS	0.000184115
69. 6.2 - M40 METAL	87. 6.2 - M24	0.000181386
69. 6.2 - M40 METAL	119. 8.9 - PRODUCTION RATIO	0.000174260
101. OFF-SITE TREATED TOTAL	108. 8.1C - OFF-SITE CONTAIN	0.000152412
91. OFF-SITE RECYCLED TOTAL	54. 5.5.1 - LANDFILLS	0.000142275
119. 8.9 - PRODUCTION RATIO	107. 8.1B - ON-SITE OTHER	0.000141690
54. 5.5.1 - LANDFILLS	119. 8.9 - PRODUCTION RATIO	0.000124778
87. 6.2 - M24	54. 5.5.1 - LANDFILLS	0.000119962
94. OFF-SITE ENERGY RECOVERY T	91. OFF-SITE RECYCLED TOTAL	0.000107352
91. OFF-SITE RECYCLED TOTAL	119. 8.9 - PRODUCTION RATIO	0.000107338
119. 8.9 - PRODUCTION RATIO	87. 6.2 - M24	0.000105923
104. TOTAL RELEASES	119. 8.9 - PRODUCTION RATIO	0.000104688
103. 6.2 - TOTAL TRANSFER	119. 8.9 - PRODUCTION RATIO	0.000098700
91. OFF-SITE RECYCLED TOTAL	112. 8.4 - RECYCLING ON SITE	0.000090500
56. 5.5.1B - OTHER LANDFILLS	87. 6.2 - M24	0.000075700
87. 6.2 - M24	109. 8.1D - OFF-SITE OTHER R	0.000073200
56. 5.5.1B - OTHER LANDFILLS	91. OFF-SITE RECYCLED TOTAL	0.000025000
103. 6.2 - TOTAL TRANSFER	56. 5.5.1B - OTHER LANDFILLS	0.000016300
91. OFF-SITE RECYCLED TOTAL	109. 8.1D - OFF-SITE OTHER R	0.000008030

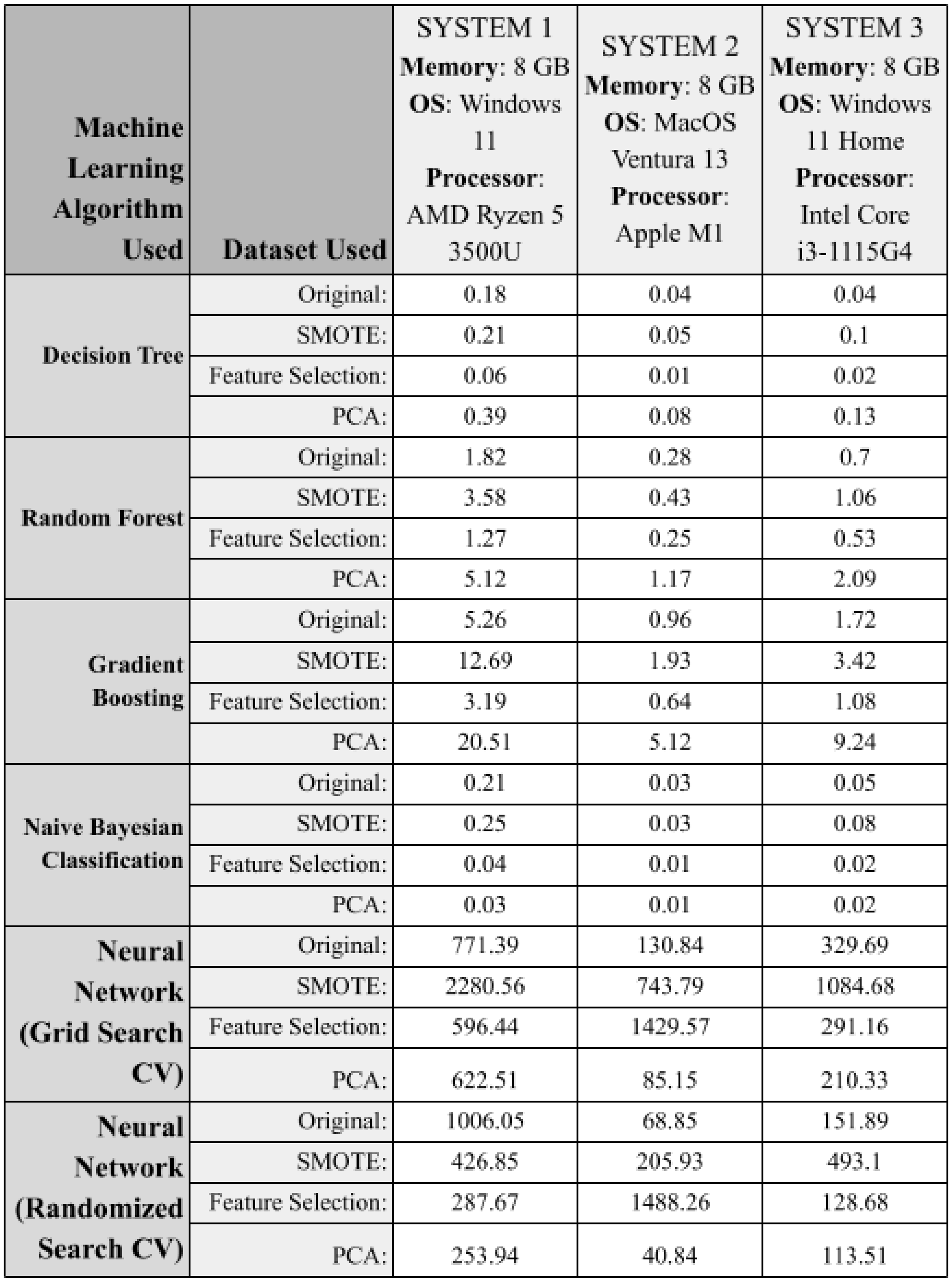
Note: *1.) Pearson correlation coefficient used. 2.) Absolute Values Taken of All Correlation Coefficients. 3.) Duplicates showing the Inverses of Two Features Have Been Removed. For Example, if A is in Column 1 and B is in Column B, then B in Column 1, A in Column 2 has been removed.*

FIGURE E: Comparison of Performance Metrics of All Machine Learning Algorithms Tested



		Dataset			
Algorithm	**Performance Metric**	**Original Dataset**	with **SMOTE** Applied	with **Feature Selection** Applied	with **PCA** Applied
Gradient Boosting	Accuracy:	0.95	0.95	0.95	0.89
	Precision:	0.89	0.6	0.9	0.43
	Recall:	0.55	0.83	0.57	0.16
	Mean Score after applying k-fold:	0.88	0.92	0.88	0.88
Neural Network with Grid Search	Accuracy:	0.87	0.87	0.89	0.88
	Precision:	0.35	0.84	0.42	0.36
	Recall:	0.24	0.91	0.23	0.18
	Mean Score after applying k-fold:	0.88	0.92	0.88	0.89
Neural Network with Randomised SearchCV	Accuracy:	0.87	0.88	0.88	0.89
	Precision:	0.21	0.93	0.29	0
	Recall:	0.08	0.83	0.12	0
	Mean Score after applying k-fold:	0.88	0.92	0.88	0.89
Decision Tree	Accuracy:	0.93	0.81	0.93	0.89
	Precision:	0.83	0.84	0.83	0.45
	Recall:	0.45	0.76	0.45	0.17
	Mean Score after applying k-fold:	0.9	0.8	0.9	0.88
Random Forest Classifier	Accuracy:	0.92	0.97	0.93	0.89
	Precision:	0.76	0.96	0.8	0.47
	Recall:	0.41	0.98	0.42	0.21
	Mean Score after applying k-fold:	0.9	0.97	0.9	0.88
Naive Bayesian Classification	Accuracy:	0.24	0.25	0.25	0.25
	Precision:	0.12	0.12	0.12	0.12
	Recall:	0.96	0.97	0.96	0.97
	Mean Score after applying k-fold:	0.26	0.58	0.26	0.24

FIGURE F: Comparison of Processing Times of All Machine Learning Algorithms Tested



Machine Learning Algorithm Used	Dataset Used	SYSTEM 1 **Memory**: 8 GB **OS**: Windows 11 **Processor**: AMD Ryzen 5 3500U	SYSTEM 2 **Memory**: 8 GB **OS**: MacOS Ventura 13 **Processor**: Apple M1	SYSTEM 3 **Memory**: 8 GB **OS**: Windows 11 Home **Processor**: Intel Core i3-1115G4
Decision Tree	Original:	0.18	0.04	0.04
	SMOTE:	0.21	0.05	0.1
	Feature Selection:	0.06	0.01	0.02
	PCA:	0.39	0.08	0.13
Random Forest	Original:	1.82	0.28	0.7
	SMOTE:	3.58	0.43	1.06
	Feature Selection:	1.27	0.25	0.53
	PCA:	5.12	1.17	2.09
Gradient Boosting	Original:	5.26	0.96	1.72
	SMOTE:	12.69	1.93	3.42
	Feature Selection:	3.19	0.64	1.08
	PCA:	20.51	5.12	9.24
Naive Bayesian Classification	Original:	0.21	0.03	0.05
	SMOTE:	0.25	0.03	0.08
	Feature Selection:	0.04	0.01	0.02
	PCA:	0.03	0.01	0.02
Neural Network (Grid Search CV)	Original:	771.39	130.84	329.69
	SMOTE:	2280.56	743.79	1084.68
	Feature Selection:	596.44	1429.57	291.16
	PCA:	622.51	85.15	210.33
Neural Network (Randomized Search CV)	Original:	1006.05	68.85	151.89
	SMOTE:	426.85	205.93	493.1
	Feature Selection:	287.67	1488.26	128.68
	PCA:	253.94	40.84	113.51

Data dictionary

See attached ‘Data Dictionary - Final.CSV’.

Code

Merging Datasets

```
#import the modules
import os
import pandas as pd
```

Washington State Cancer Registry

```
# All Cancer Data Files (must be in the same folder as notebook and) start
with 'Result'
filepaths = [f for f in os.listdir(".") if f.startswith('Result')]

# Creating Dataframe that we'll store all the Cancer Data Files from
df_append = pd.DataFrame()

#append all of the Cancer Data files together
for file in filepaths:
            df_temp = pd.read_csv(file, skiprows=4) # Data Doesn't begin for
a few rows, therefore 'skip rows'
            df_append = df_append.append(df_temp, ignore_index=True)
df_append

# Dropping the 25 rows that were for 'Washington State Cancer Incidence Data:
Washington State Department of Health, Washington State Cancer Registry,
released in April 2022'
df_append = df_append[df_append['Data Type'] == 'Incidence']

# Dropping Columns that had only contained single unique value and 95% CI
df_append.drop(['Age Group','Gender','Race', '95% CI','Data Type','Cancer
Site','Stage At Diagnosis'], axis=1, inplace=True)

# Drop Annual Population? Probably but talk to everyone...
```

TRI Preliminary Dataset

```
# All Cancer Data Files ENDS (not starts) with '_wa'
filepaths = [f for f in os.listdir(".") if f.endswith('_wa.csv')]

df_append_TRI = pd.DataFrame()
```

```
#append all files together
for file in filepaths:
            df_temp = pd.read_csv(file)
            df_append_TRI = df_append_TRI.append(df_temp, ignore_index=True)

# Geography = The Name of the County. Needs to be uppercase in order for 
merge to work
df_append['Geography'] = df_append['Geography'].str.upper()
```

Merging Data

```
# Merging Data
merged_data = pd.merge(df_append_TRI, df_append, left_on=  ['1. YEAR','7. 
COUNTY'],
                   right_on= ['Year','Geography'], how='left')
```

Value Imputation + Feature Engineering

```
# For 'Annual' Observations', 'NR' = No Record.
# df_append[df_append["Annual Observations"] == 'NR']

# As there are only four cells with this value, these values were 
individually inputed with the 'Annual Observations' for the approx value
# between the year before and after.
df_append.at[51, 'Annual Observations'] = 18
df_append.at[891, 'Annual Observations'] = 17
df_append.at[811, 'Annual Observations'] = 13
df_append.at[691, 'Annual Observations'] = 13

# Now that NR Values are removed, we can convert to Numeric..
df_append["Annual Observations"] = pd.to_numeric(df_append["Annual 
Observations"])

# ... and Calculate the Observation/Population Ratio.
df_append['Observation_Population_Ratio'] = df_append['Annual Observations'] 
/ df_append['Annual Population']

# Uncomment to Export
#merged_data.to_csv('Merged_Data.csv')
```

Main Project File

Potential To Do's...

- Might be interesting to see the relationship between carcinogen_yes and the cancer flag ratio.
- Build correlation matrix BEFORE PCA and AFTER PCA.... include this as another part of the discussion...
- add more here...

Merging/Union of Original Datasets

Jupyter Notebook is linked here.

```
# For Google Colab Only
from google.colab import files

# # Uncomment only when you want to upload a new file.
files.upload()
```

Importing Packages

```
import sys
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectFromModel
from sklearn import metrics
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.decomposition import PCA
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import RandomizedSearchCV
import time
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import GridSearchCV
from sklearn.cluster import KMeans

# Many Rows & Columns, This Enables Us to See All in Output
pd.set_option('display.max_row', None)
pd.set_option('display.max_column', None)
```

```
# Importing Data
merged_data = pd.read_csv("Merged_Data.csv")

# Changing Name so it matches original name for dataframe
df = merged_data
```

```
/usr/local/lib/python3.7/dist-packages/IPython/core/interactiveshell.py:3326:
DtypeWarning: Columns (21,22,28,29,118) have mixed types.Specify dtype option
on import or set low_memory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

```
# Creating a Table of Highest % of Missing Values and Creating a Table for
In-Class Presentation
percent_missing = merged_data.isnull().sum() * 100 / len(merged_data)
missing_value_df = pd.DataFrame({'column_name': merged_data.columns,
                                 'percent_missing': percent_missing})

missing_value_df = missing_value_df.sort_values('percent_missing',
ascending=False).head(20) # Remove Head to Show More
missing_value_df
```

<u>Data Cleaning</u>

<u>Creating 'Cancer Rate Flag' (Target Variable)</u>

```
# # UNCOMMENT TO RUN: Used This Function to Learn Our Threshold for 'Cancer
Rate' Flag

# # Function Source:
https://medium.com/@prashant.nair2050/hands-on-outlier-detection-and-treatmen
t-in-python-using-1-5-iqr-rule-f9ff1961a414
def outlier_treatment(datacolumn):
    sorted(datacolumn)
    Q1,Q3 = np.percentile(datacolumn , [25,75])
    IQR = Q3 - Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    return lower_range,upper_range

# #Getting Lower & Upperbound
lowerbound,upperbound = outlier_treatment(df['Cases_Population_Ratio'])

# # Upperbound = 0.007400788258705717. Thjis point represents the upper range
extreme outlier.
# # If below upper range extreme outlier, then the Cancer Rate of a County is
below the mean across counties. Note that this figure was calculated after
the merge. Currently our model doesn't take into account the number of
factories in a given county, so by doing this does it indirectly account for
that?
# def county_case_rate (value):
#    if value < .007400788258705717:
```

```
#       return 0
#    return 1

# df['Cancer_Rate_Flag'] = df['Cases_Population_Ratio'].map(county_case_rate)

# df['Cancer_Rate_Flag'].value_counts()

# TESTING THE 'COUNTY CASE RATE' FLAG WITH DIFFERENT VALUES...
# 0.005557 = The Mean
# 0.007400788258705717 = The Upper Range Extreme Outlier

# If below 0.007400788258705717 (upper range extreme outlier), then Cancer
Rate is below the mean across counties. Note that this figure was calculated
after the merge. CUrrently our model doesn't take into account the number of
factories in a given county, so by doing this does it indirectly account for
that?

def county_case_rate (value):
   if value < 0.007400788258705717:
      return 0
   return 1

df['Cancer_Rate_Flag'] = df['Cases_Population_Ratio'].map(county_case_rate)

df['Cancer_Rate_Flag'].value_counts()

0    24526
1     1274
Name: Cancer_Rate_Flag, dtype: int64
```

Removing & Replacing Values

Removing Irrelevant Variables

```
df.drop(['8. ST',
        '2. TRIFD',
        '5. STREET ADDRESS',
        '19. INDUSTRY SECTOR CODE',
        '22. SIC 2',
        '28. NAICS 2',
        '24. SIC 4',
        '25. SIC 5',
        '26. SIC 6',
        '27. PRIMARY NAICS',
        '21. PRIMARY SIC',
        '23. SIC 3',
        '29. NAICS 3',
        '30. NAICS 4',
        '31. NAICS 5',
        '32. NAICS 6',
        '3. FRS ID',
        '9. ZIP',
        '12. LATITUDE',
        '16. PARENT CO DB NUM',
        '13. LONGITUDE',
        '14. HORIZONTAL DATUM',
        '66. 6.2 - M10',
        '67. 6.2 - M41',
        '68. 6.2 - M62',
        '70. 6.2 - M61 METAL',
        '71. 6.2 - M71',
        '72. 6.2 - M81',
        '73. 6.2 - M82',
        '74. 6.2 - M72',
        '75. 6.2 - M63',
        '76. 6.2 - M66',
        '77. 6.2 - M67',
        '78. 6.2 - M64',
        '79. 6.2 - M65',
        '80. 6.2 - M73',
        '81. 6.2 - M79',
        '82. 6.2 - M90',
        '83. 6.2 - M94',
        '84. 6.2 - M99',
        '86. 6.2 - M20',
        '89. 6.2 - M28',
        '90. 6.2 - M93',
        '93. 6.2 - M92',
        '95. 6.2 - M40 NON-METAL',
        '97. 6.2 - M54',
        '98. 6.2 - M61 NON-METAL',
        '99. 6.2 - M69',
        '100. 6.2 - M95',
        '102. 6.2 - UNCLASSIFIED',
        '105. 8.1 - RELEASES',
        '106. 8.1A - ON-SITE CONTAINED',
        '110. 8.2 - ENERGY RECOVER ON',
        '10. BIA',
        '11. TRIBE',
        '117. 8.8 - ONE-TIME RELEASE',
        '52. 5.4.1 - UNDERGROUND CL I',
        '53. 5.4.2 - UNDERGROUND C II-V',
        '55. 5.5.1A - RCRA C LANDFILL',
        '51. 5.4 - UNDERGROUND',
        '58. 5.5.3 - SURFACE IMPNDMNT',
        '59. 5.5.3A - RCRA SURFACE IM'], axis=1, inplace=True)

# From the New Dataframe dropping the following variables due to either being
irrelevant or to avoid multicollinearity
df.drop(['Year',
'Annual Population',
'Annual Observations',
'Age-Adj. Rate per 100,000'], axis=1, inplace=True)

# From the New Dataframe dropping the following variables due to either being
irrelevant or to avoid multicollinearity
df.drop(['Geography'], axis=1, inplace=True)

# In addition, Dropping these variables
df.drop(['4. FACILITY NAME',
'6. CITY',
'15. PARENT CO NAME',
'17. STANDARD PARENT CO NAME'], axis=1, inplace=True)

# In addition, Dropping these variables
df.drop(['1. YEAR'], axis=1, inplace=True)

# Only One Value (which is 'R')
df.drop(['46. FORM TYPE'], axis=1, inplace=True)

# Droping Rows that have Grams as unit of measurement
df = df[df['47. UNIT OF MEASURE'] != 'Grams']

# Only One Value Now (which is 'Pounds')
df.drop(['47. UNIT OF MEASURE'], axis=1, inplace=True)

# Dropping '38. SRS ID' as it's the same thing just in a different format as
'37. CAS#'
df.drop(['38. SRS ID'], axis=1, inplace=True)

    # PROOF: Check Unique Value Counts for both...
    #len(pd.unique(df['38. SRS ID']))
    #len(pd.unique(df['37. CAS#']))

# Dropping as "DOC_CTRL_NUM is a unique identification number assigned to
each submission"
df.drop(['33. DOC_CTRL_NUM'], axis=1, inplace=True)

# Only One Value Now (which is 'Pounds')
df.drop(['Cases_Population_Ratio'], axis=1, inplace=True)

df.drop(['7. COUNTY'], axis=1, inplace=True)

df.drop(['Unnamed: 0'], axis=1, inplace=True)
```

```
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:4913:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy
  errors=errors,
```

Replacing Null Values with Mean

```
df['119. 8.9 - PRODUCTION RATIO'] = df['119. 8.9 - PRODUCTION
RATIO'].fillna(df['119. 8.9 - PRODUCTION RATIO'].mean())
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

Removing Rows with Null Values

```
#df.dropna(subset=['17. STANDARD PARENT CO NAME','15. PARENT CO NAME','118.
PROD_RATIO_OR_ ACTIVITY'],axis=0, how='any', inplace=True)

df.dropna(subset=['118. PROD_RATIO_OR_ ACTIVITY'],axis=0, how='any',
inplace=True)
```

```
/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returni
ng-a-view-versus-a-copy
  return func(*args, **kwargs)
```

```
# FOR VERIFICATION ONLY FOR RECHECKING NULL COUNT: Creating a Table of
Highest % of Missing Values and Creating a Table for In-Class Presentation
percent_missing = df.isnull().sum() * 100 / len(df)
missing_value_df = pd.DataFrame({'column_name': df.columns,
                                 'percent_missing': percent_missing})

missing_value_df = missing_value_df.sort_values('percent_missing',
ascending=False).head(6) # Remove Head to Show More
missing_value_df
```

```
                                         column_name  percent_missing
18. FEDERAL FACILITY            18. FEDERAL FACILITY              0.0
20. INDUSTRY SECTOR              20. INDUSTRY SECTOR              0.0
85. OFF-SITE RELEASE TOTAL    85. OFF-SITE RELEASE TOTAL          0.0
87. 6.2 - M24                          87. 6.2 - M24              0.0
```

```
88. 6.2 - M26                             88. 6.2 - M26              0.0
91. OFF-SITE RECYCLED TOTAL  91. OFF-SITE RECYCLED TOTAL             0.0
```

```
# # BELIEVE WE CAN DELETE THIS NOW...

# sns.boxplot(x=df['Cases_Population_Ratio'])
#
https://towardsdatascience.com/binning-records-on-a-continuous-variable-with-
pandas-cut-and-qcut-5d7c8e11d7b0
# 0.00764 was the number that's split point given by binning
# df['Cases_Population_Ratio_Group'] = pd.cut(df['Cases_Population_Ratio'],
2)

#
https://stackoverflow.com/questions/31511997/pandas-dataframe-replace-all-val
ues-in-a-column-based-on-condition
# df['Cases_Population_Ratio_Group'] = (df['Cases_Population_Ratio'] >
0.00764).astype(int)

# # Drop To Avoid Multicollinearity
# df.drop(['Cases_Population_Ratio_Group'], axis=1, inplace=True)
```

Correlation Matrix

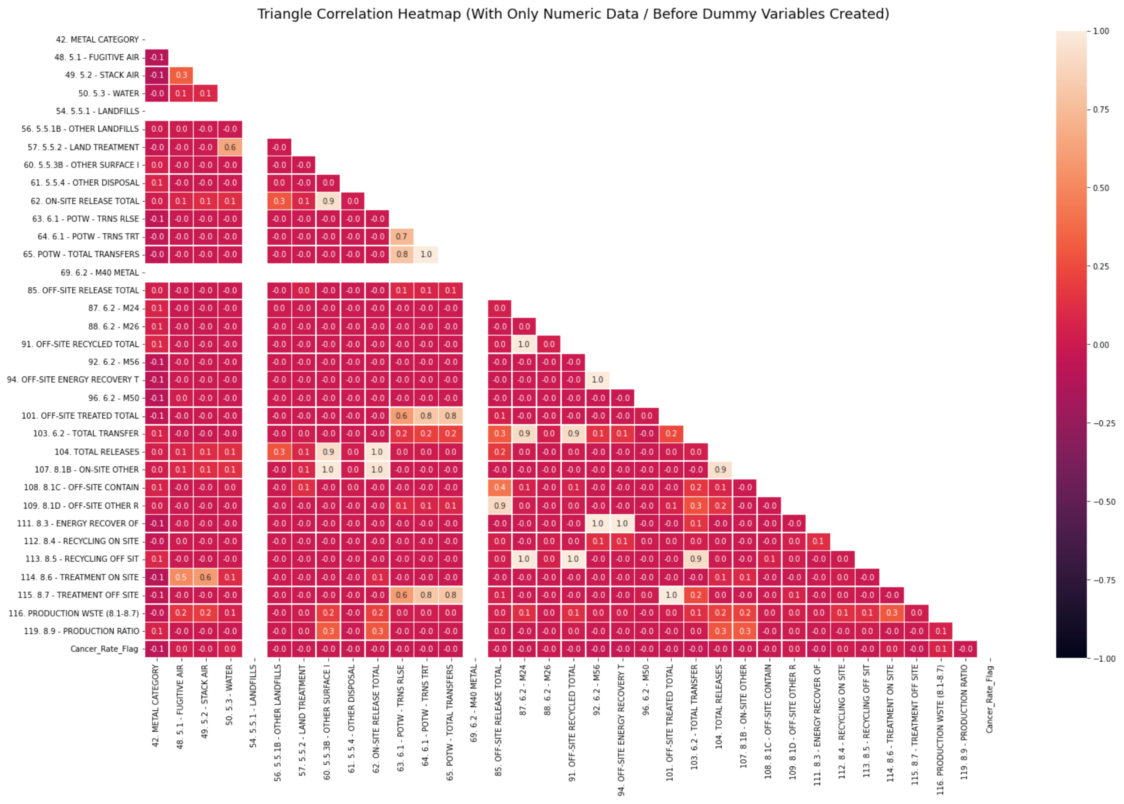
Correlation Matrix | Numeric Data Only

- Based on this, there quite a few highly correlated variables that are included before any processing such as PCA.

```
#
https://medium.com/@szabo.bibor/how-to-create-a-seaborn-correlation-heatmap-i
n-python-834c0686b88e
plt.figure(figsize=(25, 15))

# define the mask to set the values in the upper triangle to True
mask = np.triu(np.ones_like(df.corr(), dtype=np.bool))
heatmap = sns.heatmap(df.corr(),
                      mask=mask,
                      vmin=-1, vmax=1,
                      fmt=".1f", # Round to One Decimal
                      annot=True, # Annotate Values
                      #cmap='BrBG', # Change Color Palette,
                      linewidth=.5
                      )
heatmap.set_title('Triangle Correlation Heatmap (With Only Numeric Data /
Before Dummy Variables Created)', fontdict={'fontsize':18}, pad=16)
```

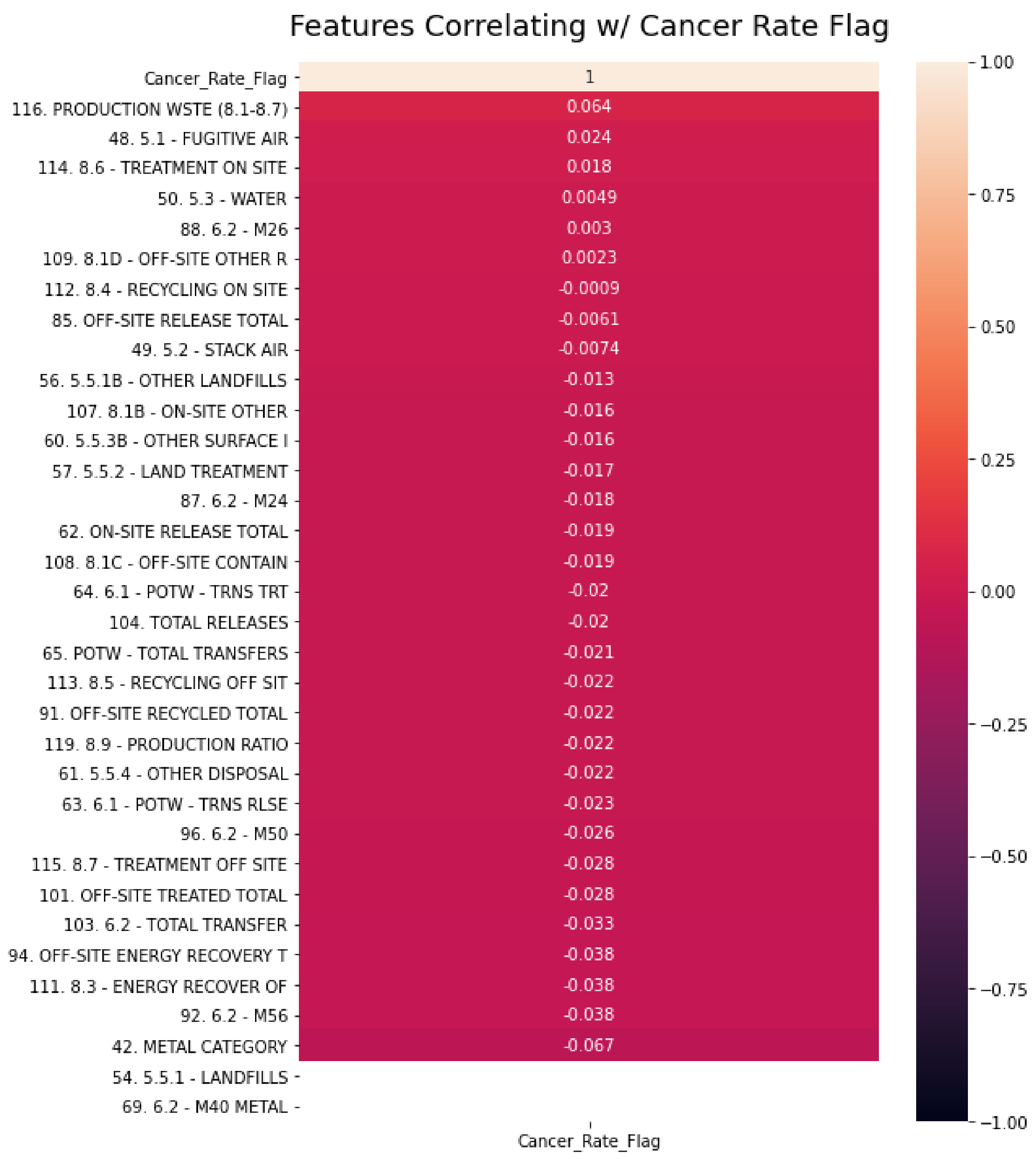
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5:
DeprecationWarning: `np.bool` is a deprecated alias for the builtin `bool`.
To silence this warning, use `bool` by itself. Doing this will not modify any
behavior and is safe. If you specifically wanted the numpy scalar type, use
```

```
`np.bool_` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  """

Text(0.5, 1.0, 'Triangle Correlation Heatmap (With Only Numeric Data / Before
Dummy Variables Created)')
```



Correlation Matrix (To Target Variable) | Numeric Data Only

```
plt.figure(figsize=(8, 12))
heatmap =
sns.heatmap(df.corr()[['Cancer_Rate_Flag']].sort_values(by='Cancer_Rate_Flag'
,

ascending=False),
                                                                  vmin=-1,
vmax=1,
                                                                  annot=True

#cmap='BrBG'
                                                                  )
heatmap.set_title('Features Correlating w/ Cancer Rate Flag',
fontdict={'fontsize':18}, pad=16)

Text(0.5, 1.0, 'Features Correlating w/ Cancer Rate Flag')
```



<u>Creating Dummy Variables</u>

- One Hot Encoding for Nominal Variables

```
dfcat_cols = df.select_dtypes('object').columns.tolist()
df = pd.get_dummies(df, columns=dfcat_cols, drop_first=True)
```

<u>Correlation Matrix (to Target Variable) Numeric AND Dummy Features</u>

- TO DO: Take this Dataframe Correlation Matrix and do it for individual Dummy Feature Groups (INDUSTRY_SECTOR, CHEMICAL, Etc.)

```
df.corr()[['Cancer_Rate_Flag']].sort_values(by='Cancer_Rate_Flag',

ascending=False)
```

```
                                                    Cancer_Rate_Flag
Cancer_Rate_Flag                                            1.000000
20. INDUSTRY SECTOR_Petroleum                               0.296383
20. INDUSTRY SECTOR_Wood Products                           0.106099
34. CHEMICAL_Hydroquinone                                   0.084388
37. CAS#_123-31-9                                           0.084388
36. TRI CHEMICAL/COMPOUND ID_0000123319                     0.084388
20. INDUSTRY SECTOR_Paper                                   0.078155
34. CHEMICAL_Hydrogen sulfide                               0.068917
36. TRI CHEMICAL/COMPOUND ID_0007783064                     0.068917
37. CAS#_7783-06-4                                          0.068917
18. FEDERAL FACILITY_YES                                    0.066618
37. CAS#_7664-93-9                                          0.065128
34. CHEMICAL_Sulfuric acid (acid aerosols inclu...          0.065128
36. TRI CHEMICAL/COMPOUND ID_0007664939                     0.065128
36. TRI CHEMICAL/COMPOUND ID_0000078795                     0.064941
37. CAS#_78-79-5                                            0.064941
34. CHEMICAL_Isoprene                                       0.064941
116. PRODUCTION WSTE (8.1-8.7)                              0.063592
37. CAS#_120-12-7                                           0.062518
36. TRI CHEMICAL/COMPOUND ID_0000120127                     0.062518
34. CHEMICAL_Anthracene                                     0.062518
36. TRI CHEMICAL/COMPOUND ID_0060207901                     0.059659
34. CHEMICAL_Propiconazole                                  0.059659
37. CAS#_60207-90-1                                         0.059659
37. CAS#_74-90-8                                            0.057541
34. CHEMICAL_Hydrogen cyanide                               0.057541
36. TRI CHEMICAL/COMPOUND ID_0000074908                     0.057541
36. TRI CHEMICAL/COMPOUND ID_0000075070                     0.056542
34. CHEMICAL_Acetaldehyde                                   0.056542
37. CAS#_75-07-0                                            0.056542
20. INDUSTRY SECTOR_Other                                   0.055052
37. CAS#_131-11-3                                           0.052808
36. TRI CHEMICAL/COMPOUND ID_0000131113                     0.052808
34. CHEMICAL_Dimethyl phthalate                             0.052808
34. CHEMICAL_Phenanthrene                                   0.049124
37. CAS#_85-01-8                                            0.049124
36. TRI CHEMICAL/COMPOUND ID_0000085018                     0.049124
37. CAS#_106-99-0                                           0.042186
36. TRI CHEMICAL/COMPOUND ID_0000106990                     0.042186
34. CHEMICAL_1,3-Butadiene                                  0.042186
34. CHEMICAL_Carbon disulfide                               0.042186
36. TRI CHEMICAL/COMPOUND ID_0000075150                     0.042186
37. CAS#_75-15-0                                            0.042186
34. CHEMICAL_Methanol                                       0.039253
37. CAS#_67-56-1                                            0.039253
36. TRI CHEMICAL/COMPOUND ID_0000067561                     0.039253
36. TRI CHEMICAL/COMPOUND ID_0000074851                     0.039129
37. CAS#_74-85-1                                            0.039129
34. CHEMICAL_Ethylene                                       0.039129
```

```
37. CAS#_463-58-1                                         0.039129
36. TRI CHEMICAL/COMPOUND ID_0000463581                   0.039129
34. CHEMICAL_Carbonyl sulfide                             0.039129
37. CAS#_7439-97-6                                        0.037509
34. CHEMICAL_Mercury                                      0.037509
36. TRI CHEMICAL/COMPOUND ID_0007439976                   0.037509
37. CAS#_79-21-0                                          0.037509
34. CHEMICAL_Peracetic acid                               0.037509
36. TRI CHEMICAL/COMPOUND ID_0000079210                   0.037509
36. TRI CHEMICAL/COMPOUND ID_0001336363                   0.037325
37. CAS#_1336-36-3                                        0.037325
34. CHEMICAL_Polychlorinated biphenyls                    0.037325
34. CHEMICAL_Propionaldehyde                              0.037325
37. CAS#_123-38-6                                         0.037325
36. TRI CHEMICAL/COMPOUND ID_0000123386                   0.037325
37. CAS#_50-00-0                                          0.036528
34. CHEMICAL_Formaldehyde                                 0.036528
34. CHEMICAL_Cresol (mixed isomers)                       0.036293
37. CAS#_1319-77-3                                        0.036293
36. TRI CHEMICAL/COMPOUND ID_0001319773                   0.036293
37. CAS#_872-50-4                                         0.033858
34. CHEMICAL_N-Methyl-2-pyrrolidone                       0.033858
36. TRI CHEMICAL/COMPOUND ID_0000872504                   0.033858
34. CHEMICAL_Propylene                                    0.033650
36. TRI CHEMICAL/COMPOUND ID_0000115071                   0.033650
37. CAS#_115-07-1                                         0.033650
34. CHEMICAL_Phenol                                       0.033383
36. TRI CHEMICAL/COMPOUND ID_0000108952                   0.033383
37. CAS#_108-95-2                                         0.033383
34. CHEMICAL_Vanadium compounds                           0.032129
37. CAS#_N770                                             0.032129
36. TRI CHEMICAL/COMPOUND ID_N770                         0.032129
36. TRI CHEMICAL/COMPOUND ID_0000100425                   0.028844
37. CAS#_100-42-5                                         0.028844
34. CHEMICAL_Styrene                                      0.028844
34. CHEMICAL_Barium  And Barium Compounds                 0.026506
37. CAS#_120-80-9                                         0.024955
36. TRI CHEMICAL/COMPOUND ID_0000120809                   0.024955
34. CHEMICAL_Catechol                                     0.024955
36. TRI CHEMICAL/COMPOUND ID_0000098828                   0.024566
34. CHEMICAL_Cumene                                       0.024566
37. CAS#_98-82-8                                          0.024566
48. 5.1 - FUGITIVE AIR                                    0.024089
34. CHEMICAL_Biphenyl                                     0.023750
37. CAS#_92-52-4                                          0.023750
36. TRI CHEMICAL/COMPOUND ID_0000092524                   0.023750
34. CHEMICAL_Molybdenum trioxide                          0.023750
37. CAS#_1313-27-5                                        0.023750
36. TRI CHEMICAL/COMPOUND ID_0001313275                   0.023750
43. CARCINOGEN_YES                                        0.021841
```

```
34. CHEMICAL_Cyclohexane                                0.020707
37. CAS#_110-82-7                                       0.020707
36. TRI CHEMICAL/COMPOUND ID_0000110827                 0.020707
34. CHEMICAL_Ammonia                                    0.020604
37. CAS#_7664-41-7                                      0.020604
36. TRI CHEMICAL/COMPOUND ID_0007664417                 0.020604
34. CHEMICAL_Nickel compounds                           0.020101
34. CHEMICAL_Manganese  And Manganese Compounds         0.019516
114. 8.6 - TREATMENT ON SITE                            0.017919
34. CHEMICAL_Tetrachloroethylene                        0.017332
37. CAS#_127-18-4                                       0.017332
36. TRI CHEMICAL/COMPOUND ID_0000127184                 0.017332
36. TRI CHEMICAL/COMPOUND ID_N590                       0.014785
34. CHEMICAL_Polycyclic aromatic compounds              0.014785
37. CAS#_N590                                           0.014785
34. CHEMICAL_Cobalt compounds                           0.014378
37. CAS#_N495                                           0.012067
36. TRI CHEMICAL/COMPOUND ID_N495                       0.012067
37. CAS#_N096                                           0.010315
36. TRI CHEMICAL/COMPOUND ID_N096                       0.010315
36. TRI CHEMICAL/COMPOUND ID_0000111422                 0.009838
37. CAS#_111-42-2                                       0.009838
34. CHEMICAL_Diethanolamine                             0.009838
34. CHEMICAL_Lead                                       0.006307
37. CAS#_7439-92-1                                      0.006307
36. TRI CHEMICAL/COMPOUND ID_0007439921                 0.006307
34. CHEMICAL_Diisocyanates                              0.005754
36. TRI CHEMICAL/COMPOUND ID_N120                       0.005754
37. CAS#_N120                                           0.005754
34. CHEMICAL_Hydrochloric acid (acid aerosols i...      0.005534
37. CAS#_7647-01-0                                      0.005534
36. TRI CHEMICAL/COMPOUND ID_0007647010                 0.005534
40. CLASSIFICATION_TRI                                  0.005124
50. 5.3 - WATER                                         0.004914
39. CLEAN AIR ACT CHEMICAL_YES                          0.003799
88. 6.2 - M26                                           0.003042
109. 8.1D - OFF-SITE OTHER R                            0.002275
112. 8.4 - RECYCLING ON SITE                           -0.000898
36. TRI CHEMICAL/COMPOUND ID_0001330207                -0.000907
37. CAS#_1330-20-7                                     -0.000907
34. CHEMICAL_Xylene (mixed isomers)                    -0.000907
34. CHEMICAL_Copper compounds                          -0.001264
20. INDUSTRY SECTOR_Machinery                          -0.002529
34. CHEMICAL_Lead  And Lead Compounds                  -0.003931
44. PBT_YES                                            -0.005124
85. OFF-SITE RELEASE TOTAL                             -0.006102
36. TRI CHEMICAL/COMPOUND ID_N450                      -0.006559
37. CAS#_N450                                          -0.006559
36. TRI CHEMICAL/COMPOUND ID_0007440666                -0.006884
34. CHEMICAL_Zinc (fume or dust)                       -0.006884
```

```
37. CAS#_7440-66-6                                         -0.006884
34. CHEMICAL_Cobalt  And Cobalt Compounds                  -0.006884
37. CAS#_N020                                              -0.006884
36. TRI CHEMICAL/COMPOUND ID_N020                          -0.006884
34. CHEMICAL_Arsenic compounds                             -0.006884
37. CAS#_79-01-6                                           -0.006884
37. CAS#_121-44-8                                          -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000108383                    -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000079016                    -0.006884
34. CHEMICAL_Trichloroethylene                             -0.006884
34. CHEMICAL_Triethylamine                                 -0.006884
37. CAS#_108-38-3                                          -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000121448                    -0.006884
37. CAS#_71-55-6                                           -0.006884
34. CHEMICAL_m-Xylene                                      -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000071556                    -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000106423                    -0.006884
37. CAS#_106-42-3                                          -0.006884
34. CHEMICAL_p-Xylene                                      -0.006884
37. CAS#_79-06-1                                           -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000079061                    -0.006884
34. CHEMICAL_Acrylamide                                    -0.006884
34. CHEMICAL_o-Xylene                                      -0.006884
37. CAS#_95-47-6                                           -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000095476                    -0.006884
34. CHEMICAL_Cadmium compounds                             -0.006884
36. TRI CHEMICAL/COMPOUND ID_0000076062                    -0.006884
36. TRI CHEMICAL/COMPOUND ID_0010061026                    -0.006884
34. CHEMICAL_Chloropicrin                                  -0.006884
37. CAS#_N078                                              -0.006884
34. CHEMICAL_trans-1,3-Dichloropropene                     -0.006884
37. CAS#_76-06-2                                           -0.006884
36. TRI CHEMICAL/COMPOUND ID_N078                          -0.006884
37. CAS#_10061-02-6                                        -0.006884
49. 5.2 - STACK AIR                                        -0.007437
37. CAS#_110-54-3                                          -0.007513
36. TRI CHEMICAL/COMPOUND ID_0000110543                    -0.007513
34. CHEMICAL_n-Hexane                                      -0.007513
37. CAS#_N040                                              -0.007994
36. TRI CHEMICAL/COMPOUND ID_N040                          -0.007994
34. CHEMICAL_Ethylbenzene                                  -0.008448
36. TRI CHEMICAL/COMPOUND ID_0000100414                    -0.008448
36. TRI CHEMICAL/COMPOUND ID_N100                          -0.009121
37. CAS#_N100                                              -0.009121
36. TRI CHEMICAL/COMPOUND ID_0000191242                    -0.009364
34. CHEMICAL_Benzo[g,h,i]perylene                          -0.009364
37. CAS#_191-24-2                                          -0.009364
34. CHEMICAL_Benzoyl peroxide                              -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000094360                    -0.009737
37. CAS#_94-36-0                                           -0.009737
```

```
37. CAS#_123-91-1                                      -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000123911                -0.009737
34. CHEMICAL_1,4-Dioxane                               -0.009737
34. CHEMICAL_Dicyclopentadiene                         -0.009737
37. CAS#_77-73-6                                       -0.009737
34. CHEMICAL_tert-Butyl alcohol                        -0.009737
37. CAS#_75-65-0                                       -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000075650                -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000077736                -0.009737
37. CAS#_74-87-3                                       -0.009737
37. CAS#_78-92-2                                       -0.009737
34. CHEMICAL_sec-Butyl alcohol                         -0.009737
36. TRI CHEMICAL/COMPOUND ID_N106                      -0.009737
37. CAS#_N106                                          -0.009737
34. CHEMICAL_Cyanide compounds                         -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000074873                -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000078922                -0.009737
34. CHEMICAL_Chloromethane                             -0.009737
37. CAS#_554-13-2                                      -0.009737
36. TRI CHEMICAL/COMPOUND ID_0000554132                -0.009737
34. CHEMICAL_Lithium carbonate                         -0.009737
20. INDUSTRY SECTOR_Textiles                           -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000122394                -0.011928
34. CHEMICAL_Diphenylamine                             -0.011928
37. CAS#_122-39-4                                      -0.011928
34. CHEMICAL_1,3-Dichloropropylene                     -0.011928
37. CAS#_542-75-6                                      -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000542756                -0.011928
37. CAS#_75-09-2                                       -0.011928
34. CHEMICAL_Chloroform                                -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000075092                -0.011928
34. CHEMICAL_Dichloromethane                           -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000067663                -0.011928
37. CAS#_67-66-3                                       -0.011928
37. CAS#_68-12-2                                       -0.011928
34. CHEMICAL_Acetonitrile                              -0.011928
37. CAS#_7440-36-0                                     -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000075058                -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000068122                -0.011928
34. CHEMICAL_N,N-Dimethylformamide                     -0.011928
37. CAS#_75-05-8                                       -0.011928
36. TRI CHEMICAL/COMPOUND ID_0007440360                -0.011928
34. CHEMICAL_Antimony                                  -0.011928
34. CHEMICAL_1,2-Dichloroethane                        -0.011928
36. TRI CHEMICAL/COMPOUND ID_0000107062                -0.011928
37. CAS#_107-06-2                                      -0.011928
37. CAS#_25321-14-6                                    -0.011928
36. TRI CHEMICAL/COMPOUND ID_0007440484                -0.011928
36. TRI CHEMICAL/COMPOUND ID_0025321146                -0.011928
37. CAS#_7440-22-4                                     -0.011928
```

34. CHEMICAL_Silver	-0.011928
36. TRI CHEMICAL/COMPOUND ID_0007440224	-0.011928
37. CAS#_7440-48-4	-0.011928
34. CHEMICAL_Dinitrotoluene (mixed isomers)	-0.011928
34. CHEMICAL_Cobalt	-0.011928
34. CHEMICAL_Pyridine	-0.011928
36. TRI CHEMICAL/COMPOUND ID_0000110861	-0.011928
37. CAS#_110-86-1	-0.011928
37. CAS#_137-41-7	-0.011928
36. TRI CHEMICAL/COMPOUND ID_0000137417	-0.011928
34. CHEMICAL_Potassium N-methyldithiocarbamate	-0.011928
37. CAS#_1634-04-4	-0.011928
34. CHEMICAL_Methyl tert-butyl ether	-0.011928
36. TRI CHEMICAL/COMPOUND ID_0001634044	-0.011928
35. ELEMENTAL METAL INCLUDED_YES	-0.012012
34. CHEMICAL_Manganese compounds	-0.013008
56. 5.5.1B - OTHER LANDFILLS	-0.013256
34. CHEMICAL_Benzene	-0.013700
36. TRI CHEMICAL/COMPOUND ID_0000071432	-0.013700
37. CAS#_71-43-2	-0.013700
36. TRI CHEMICAL/COMPOUND ID_0000087865	-0.013776
37. CAS#_87-86-5	-0.013776
34. CHEMICAL_Pentachlorophenol	-0.013776
36. TRI CHEMICAL/COMPOUND ID_0000124403	-0.013776
37. CAS#_124-40-3	-0.013776
34. CHEMICAL_Dimethylamine	-0.013776
36. TRI CHEMICAL/COMPOUND ID_0000055630	-0.013776
34. CHEMICAL_Mercury And Mercury Compounds	-0.013776
37. CAS#_55-63-0	-0.013776
34. CHEMICAL_Nitroglycerin	-0.013776
34. CHEMICAL_Nickel And Nickel Compounds	-0.013776
34. CHEMICAL_Lead compounds	-0.014001
34. CHEMICAL_Mercury compounds	-0.014523
36. TRI CHEMICAL/COMPOUND ID_0000095636	-0.014523
37. CAS#_95-63-6	-0.014523
34. CHEMICAL_1,2,4-Trimethylbenzene	-0.014523
36. TRI CHEMICAL/COMPOUND ID_N420	-0.014631
37. CAS#_N420	-0.014631
36. TRI CHEMICAL/COMPOUND ID_0000071363	-0.014794
34. CHEMICAL_n-Butyl alcohol	-0.014794
37. CAS#_71-36-3	-0.014794
37. CAS#_80-62-6	-0.015405
34. CHEMICAL_Methyl methacrylate	-0.015405
36. TRI CHEMICAL/COMPOUND ID_0000080626	-0.015405
107. 8.1B - ON-SITE OTHER	-0.015841
60. 5.5.3B - OTHER SURFACE I	-0.015909
34. CHEMICAL_Toluene diisocyanate (mixed isomers)	-0.016879
37. CAS#_26471-62-5	-0.016879
36. TRI CHEMICAL/COMPOUND ID_0026471625	-0.016879
36. TRI CHEMICAL/COMPOUND ID_0000117817	-0.016879

```
37. CAS#_117-81-7                                          -0.016879
34. CHEMICAL_Chromium  and Chromium Compounds(e...         -0.016879
34. CHEMICAL_Di(2-ethylhexyl) phthalate                    -0.016879
36. TRI CHEMICAL/COMPOUND ID_0000137428                    -0.016879
37. CAS#_137-42-8                                          -0.016879
34. CHEMICAL_Metham sodium                                 -0.016879
57. 5.5.2 - LAND TREATMENT                                 -0.017118
37. CAS#_N458                                              -0.017683
36. TRI CHEMICAL/COMPOUND ID_N458                          -0.017683
20. INDUSTRY SECTOR_Chemicals                              -0.017786
34. CHEMICAL_Copper  And Copper Compounds                  -0.018235
87. 6.2 - M24                                              -0.018354
62. ON-SITE RELEASE TOTAL                                  -0.019068
108. 8.1C - OFF-SITE CONTAIN                               -0.019191
34. CHEMICAL_Toluene                                       -0.019193
36. TRI CHEMICAL/COMPOUND ID_0000108883                    -0.019193
37. CAS#_108-88-3                                          -0.019193
36. TRI CHEMICAL/COMPOUND ID_0000106945                    -0.019498
34. CHEMICAL_1-Bromopropane                                -0.019498
37. CAS#_106-94-5                                          -0.019498
37. CAS#_79-94-7                                           -0.019498
36. TRI CHEMICAL/COMPOUND ID_0000079947                    -0.019498
34. CHEMICAL_Tetrabromobisphenol A                         -0.019498
64. 6.1 - POTW - TRNS TRT                                  -0.019647
104. TOTAL RELEASES                                        -0.019873
37. CAS#_10049-04-4                                        -0.020685
36. TRI CHEMICAL/COMPOUND ID_0010049044                    -0.020685
34. CHEMICAL_Chlorine dioxide                              -0.020685
65. POTW - TOTAL TRANSFERS                                 -0.020846
113. 8.5 - RECYCLING OFF SIT                               -0.021548
91. OFF-SITE RECYCLED TOTAL                                -0.021551
119. 8.9 - PRODUCTION RATIO                                -0.021989
61. 5.5.4 - OTHER DISPOSAL                                 -0.022140
34. CHEMICAL_Barium compounds (except for bariu...         -0.022878
20. INDUSTRY SECTOR_Electrical Equipment                   -0.022878
63. 6.1 - POTW - TRNS RLSE                                 -0.023353
36. TRI CHEMICAL/COMPOUND ID_0007782505                    -0.023900
34. CHEMICAL_Chlorine                                      -0.023900
37. CAS#_7782-50-5                                         -0.023900
34. CHEMICAL_Chromium compounds (except for chr...         -0.025708
96. 6.2 - M50                                              -0.026480
37. CAS#_91-20-3                                           -0.026803
34. CHEMICAL_Naphthalene                                   -0.026803
36. TRI CHEMICAL/COMPOUND ID_0000091203                    -0.026803
37. CAS#_64-18-6                                           -0.027620
36. TRI CHEMICAL/COMPOUND ID_0000064186                    -0.027620
34. CHEMICAL_Formic acid                                   -0.027620
115. 8.7 - TREATMENT OFF SITE                              -0.027696
101. OFF-SITE TREATED TOTAL                                -0.027697
34. CHEMICAL_Methyl isobutyl ketone                        -0.029308
```

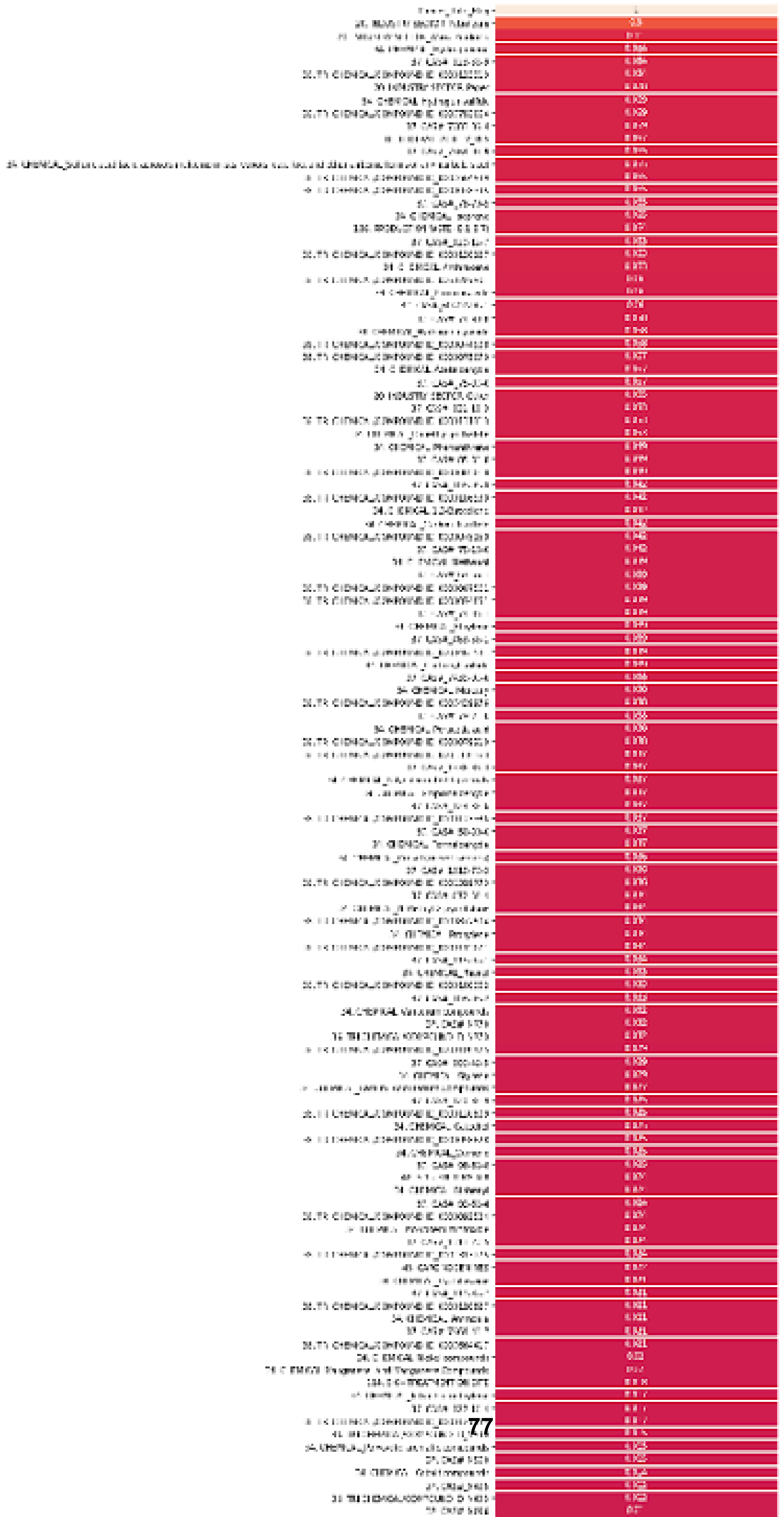
```
36. TRI CHEMICAL/COMPOUND ID_0000108101                    -0.029308
37. CAS#_108-10-1                                          -0.029308
20. INDUSTRY SECTOR_Computers and Electronic Pr...         -0.029308
36. TRI CHEMICAL/COMPOUND ID_N090                          -0.030431
37. CAS#_N090                                              -0.030431
36. TRI CHEMICAL/COMPOUND ID_0000107211                    -0.030906
34. CHEMICAL_Ethylene glycol                               -0.030906
37. CAS#_107-21-1                                          -0.030906
37. CAS#_N511                                              -0.031648
36. TRI CHEMICAL/COMPOUND ID_N511                          -0.031648
34. CHEMICAL_Nitrate compounds (water dissociab...         -0.031648
103. 6.2 - TOTAL TRANSFER                                  -0.033052
34. CHEMICAL_Certain glycol ethers                         -0.034590
37. CAS#_N230                                              -0.034590
36. TRI CHEMICAL/COMPOUND ID_N230                          -0.034590
34. CHEMICAL_Hydrogen fluoride                             -0.035961
37. CAS#_7664-39-3                                         -0.035961
36. TRI CHEMICAL/COMPOUND ID_0007664393                    -0.035961
94. OFF-SITE ENERGY RECOVERY T                             -0.037819
111. 8.3 - ENERGY RECOVER OF                               -0.037819
37. CAS#_7439-96-5                                         -0.037845
36. TRI CHEMICAL/COMPOUND ID_0007439965                    -0.037845
34. CHEMICAL_Manganese                                     -0.037845
92. 6.2 - M56                                              -0.037922
37. CAS#_7440-50-8                                         -0.039666
36. TRI CHEMICAL/COMPOUND ID_0007440508                    -0.039666
34. CHEMICAL_Copper                                        -0.039666
20. INDUSTRY SECTOR_Metal Mining                           -0.039807
37. CAS#_N982                                              -0.039807
36. TRI CHEMICAL/COMPOUND ID_N982                          -0.039807
34. CHEMICAL_Zinc compounds                                -0.039807
36. TRI CHEMICAL/COMPOUND ID_0007440020                    -0.040257
37. CAS#_7440-02-0                                         -0.040257
34. CHEMICAL_Nickel                                        -0.040257
36. TRI CHEMICAL/COMPOUND ID_0007440473                    -0.044748
37. CAS#_7440-47-3                                         -0.044748
34. CHEMICAL_Chromium                                      -0.044748
20. INDUSTRY SECTOR_Electric Utilities                     -0.044992
20. INDUSTRY SECTOR_Plastics and Rubber                    -0.047125
20. INDUSTRY SECTOR_Food                                   -0.053154
36. TRI CHEMICAL/COMPOUND ID_0007697372                    -0.057125
37. CAS#_7697-37-2                                         -0.057125
34. CHEMICAL_Nitric acid                                   -0.057125
20. INDUSTRY SECTOR_Nonmetallic Mineral Product            -0.066850
42. METAL CATEGORY                                         -0.066917
20. INDUSTRY SECTOR_Hazardous Waste                        -0.067296
41. METAL_YES                                              -0.074891
20. INDUSTRY SECTOR_Fabricated Metals                      -0.082278
20. INDUSTRY SECTOR_Primary Metals                         -0.095661
20. INDUSTRY SECTOR_Transportation Equipment               -0.096744
```

```
20. INDUSTRY SECTOR_Petroleum Bulk Terminals                  -0.097106
118. PROD_RATIO_OR_ ACTIVITY_PRODUCTION                       -0.201696
54. 5.5.1 - LANDFILLS                                               NaN
69. 6.2 - M40 METAL                                                 NaN
```

```
plt.figure(figsize=(8, 120))
heatmap =
sns.heatmap(df.corr()[['Cancer_Rate_Flag']].sort_values(by='Cancer_Rate_Flag'
,

ascending=False),
                                                                 vmin=-1,
vmax=1,
                                                                 annot=True,

linewidth=.1

#cmap='BrBG'
                                                                 )
heatmap.set_title('All Numeric + Dummy Features Correlating w/ Cancer Rate
Flag', fontdict={'fontsize':18}, pad=16)
```

```
Text(0.5, 1.0, 'All Numeric + Dummy Features Correlating w/ Cancer Rate
Flag')
```

All Numeric + Dummy Features Correlating w/ Cancer Rate Flag



Currently Have this setup as absolute value. Should we remove?

- In our appendix, we could have the correlation matrix as seen here which includes all variables, BUT we could also have ones where we only include features that were part of the same column before dummies. For example, we create a table (or chart) for just INDUSTRY SECTOR. Could do this easily in Excel.

```
# Correlation of Values To Target Variable Sorted Highest to Lowest
ab_col_matrix = abs(df.corr())
ab_col_matrix['Cancer_Rate_Flag'].sort_values(ascending=False)
```

```
Cancer_Rate_Flag
1.000000
20. INDUSTRY SECTOR_Petroleum
0.296383
118. PROD_RATIO_OR_ ACTIVITY_PRODUCTION
0.201696
20. INDUSTRY SECTOR_Wood Products
0.106099
20. INDUSTRY SECTOR_Petroleum Bulk Terminals
0.097106
20. INDUSTRY SECTOR_Transportation Equipment
0.096744
20. INDUSTRY SECTOR_Primary Metals
0.095661
37. CAS#_123-31-9
0.084388
36. TRI CHEMICAL/COMPOUND ID_0000123319
0.084388
34. CHEMICAL_Hydroquinone
0.084388
20. INDUSTRY SECTOR_Fabricated Metals
0.082278
20. INDUSTRY SECTOR_Paper
0.078155
41. METAL_YES
0.074891
36. TRI CHEMICAL/COMPOUND ID_0007783064
0.068917
37. CAS#_7783-06-4
0.068917
34. CHEMICAL_Hydrogen sulfide
0.068917
20. INDUSTRY SECTOR_Hazardous Waste
0.067296
42. METAL CATEGORY
0.066917
20. INDUSTRY SECTOR_Nonmetallic Mineral Product
0.066850
18. FEDERAL FACILITY_YES
0.066618
```

```
36. TRI CHEMICAL/COMPOUND ID_0007664939
0.065128
37. CAS#_7664-93-9
0.065128
34. CHEMICAL_Sulfuric acid (acid aerosols including mists, vapors, gas, fog,
and other airborne forms of any particle size)        0.065128
36. TRI CHEMICAL/COMPOUND ID_0000078795
0.064941
34. CHEMICAL_Isoprene
0.064941
37. CAS#_78-79-5
0.064941
116. PRODUCTION WSTE (8.1-8.7)
0.063592
36. TRI CHEMICAL/COMPOUND ID_0000120127
0.062518
34. CHEMICAL_Anthracene
0.062518
37. CAS#_120-12-7
0.062518
37. CAS#_60207-90-1
0.059659
34. CHEMICAL_Propiconazole
0.059659
36. TRI CHEMICAL/COMPOUND ID_0060207901
0.059659
36. TRI CHEMICAL/COMPOUND ID_0000074908
0.057541
34. CHEMICAL_Hydrogen cyanide
0.057541
37. CAS#_74-90-8
0.057541
37. CAS#_7697-37-2
0.057125
36. TRI CHEMICAL/COMPOUND ID_0007697372
0.057125
34. CHEMICAL_Nitric acid
0.057125
34. CHEMICAL_Acetaldehyde
0.056542
36. TRI CHEMICAL/COMPOUND ID_0000075070
0.056542
37. CAS#_75-07-0
0.056542
20. INDUSTRY SECTOR_Other
0.055052
20. INDUSTRY SECTOR_Food
0.053154
34. CHEMICAL_Dimethyl phthalate
0.052808
```

```
36. TRI CHEMICAL/COMPOUND ID_0000131113
0.052808
37. CAS#_131-11-3
0.052808
37. CAS#_85-01-8
0.049124
36. TRI CHEMICAL/COMPOUND ID_0000085018
0.049124
34. CHEMICAL_Phenanthrene
0.049124
20. INDUSTRY SECTOR_Plastics and Rubber
0.047125
20. INDUSTRY SECTOR_Electric Utilities
0.044992
34. CHEMICAL_Chromium
0.044748
37. CAS#_7440-47-3
0.044748
36. TRI CHEMICAL/COMPOUND ID_0007440473
0.044748
37. CAS#_106-99-0
0.042186
34. CHEMICAL_1,3-Butadiene
0.042186
36. TRI CHEMICAL/COMPOUND ID_0000106990
0.042186
36. TRI CHEMICAL/COMPOUND ID_0000075150
0.042186
34. CHEMICAL_Carbon disulfide
0.042186
37. CAS#_75-15-0
0.042186
34. CHEMICAL_Nickel
0.040257
36. TRI CHEMICAL/COMPOUND ID_0007440020
0.040257
37. CAS#_7440-02-0
0.040257
34. CHEMICAL_Zinc compounds
0.039807
36. TRI CHEMICAL/COMPOUND ID_N982
0.039807
37. CAS#_N982
0.039807
20. INDUSTRY SECTOR_Metal Mining
0.039807
34. CHEMICAL_Copper
0.039666
36. TRI CHEMICAL/COMPOUND ID_0007440508
0.039666
```

```
37. CAS#_7440-50-8
0.039666
37. CAS#_67-56-1
0.039253
34. CHEMICAL_Methanol
0.039253
36. TRI CHEMICAL/COMPOUND ID_0000067561
0.039253
36. TRI CHEMICAL/COMPOUND ID_0000074851
0.039129
37. CAS#_74-85-1
0.039129
34. CHEMICAL_Ethylene
0.039129
34. CHEMICAL_Carbonyl sulfide
0.039129
36. TRI CHEMICAL/COMPOUND ID_0000463581
0.039129
37. CAS#_463-58-1
0.039129
92. 6.2 - M56
0.037922
34. CHEMICAL_Manganese
0.037845
37. CAS#_7439-96-5
0.037845
36. TRI CHEMICAL/COMPOUND ID_0007439965
0.037845
94. OFF-SITE ENERGY RECOVERY T
0.037819
111. 8.3 - ENERGY RECOVER OF
0.037819
37. CAS#_7439-97-6
0.037509
34. CHEMICAL_Mercury
0.037509
36. TRI CHEMICAL/COMPOUND ID_0007439976
0.037509
36. TRI CHEMICAL/COMPOUND ID_0000079210
0.037509
37. CAS#_79-21-0
0.037509
34. CHEMICAL_Peracetic acid
0.037509
36. TRI CHEMICAL/COMPOUND ID_0001336363
0.037325
37. CAS#_1336-36-3
0.037325
34. CHEMICAL_Polychlorinated biphenyls
0.037325
```

```
36. TRI CHEMICAL/COMPOUND ID_0000123386
0.037325
34. CHEMICAL_Propionaldehyde
0.037325
37. CAS#_123-38-6
0.037325
34. CHEMICAL_Formaldehyde
0.036528
37. CAS#_50-00-0
0.036528
37. CAS#_1319-77-3
0.036293
36. TRI CHEMICAL/COMPOUND ID_0001319773
0.036293
34. CHEMICAL_Cresol (mixed isomers)
0.036293
34. CHEMICAL_Hydrogen fluoride
0.035961
37. CAS#_7664-39-3
0.035961
36. TRI CHEMICAL/COMPOUND ID_0007664393
0.035961
37. CAS#_N230
0.034590
34. CHEMICAL_Certain glycol ethers
0.034590
36. TRI CHEMICAL/COMPOUND ID_N230
0.034590
34. CHEMICAL_N-Methyl-2-pyrrolidone
0.033858
36. TRI CHEMICAL/COMPOUND ID_0000872504
0.033858
37. CAS#_872-50-4
0.033858
34. CHEMICAL_Propylene
0.033650
36. TRI CHEMICAL/COMPOUND ID_0000115071
0.033650
37. CAS#_115-07-1
0.033650
37. CAS#_108-95-2
0.033383
34. CHEMICAL_Phenol
0.033383
36. TRI CHEMICAL/COMPOUND ID_0000108952
0.033383
103. 6.2 - TOTAL TRANSFER
0.033052
37. CAS#_N770
0.032129
```

```
36. TRI CHEMICAL/COMPOUND ID_N770
0.032129
34. CHEMICAL_Vanadium compounds
0.032129
37. CAS#_N511
0.031648
36. TRI CHEMICAL/COMPOUND ID_N511
0.031648
34. CHEMICAL_Nitrate compounds (water dissociable; reportable only when in
aqueous solution)                                      0.031648
37. CAS#_107-21-1
0.030906
36. TRI CHEMICAL/COMPOUND ID_0000107211
0.030906
34. CHEMICAL_Ethylene glycol
0.030906
36. TRI CHEMICAL/COMPOUND ID_N090
0.030431
37. CAS#_N090
0.030431
20. INDUSTRY SECTOR_Computers and Electronic Products
0.029308
34. CHEMICAL_Methyl isobutyl ketone
0.029308
37. CAS#_108-10-1
0.029308
36. TRI CHEMICAL/COMPOUND ID_0000108101
0.029308
37. CAS#_100-42-5
0.028844
34. CHEMICAL_Styrene
0.028844
36. TRI CHEMICAL/COMPOUND ID_0000100425
0.028844
101. OFF-SITE TREATED TOTAL
0.027697
115. 8.7 - TREATMENT OFF SITE
0.027696
36. TRI CHEMICAL/COMPOUND ID_0000064186
0.027620
37. CAS#_64-18-6
0.027620
34. CHEMICAL_Formic acid
0.027620
34. CHEMICAL_Naphthalene
0.026803
37. CAS#_91-20-3
0.026803
36. TRI CHEMICAL/COMPOUND ID_0000091203
0.026803
```

```
34. CHEMICAL_Barium  And Barium Compounds
0.026506
96. 6.2 - M50
0.026480
34. CHEMICAL_Chromium compounds (except for chromite ore mined in the
Transvaal Region)                                            0.025708
37. CAS#_120-80-9
0.024955
34. CHEMICAL_Catechol
0.024955
36. TRI CHEMICAL/COMPOUND ID_0000120809
0.024955
36. TRI CHEMICAL/COMPOUND ID_0000098828
0.024566
34. CHEMICAL_Cumene
0.024566
37. CAS#_98-82-8
0.024566
48. 5.1 - FUGITIVE AIR
0.024089
36. TRI CHEMICAL/COMPOUND ID_0007782505
0.023900
34. CHEMICAL_Chlorine
0.023900
37. CAS#_7782-50-5
0.023900
34. CHEMICAL_Biphenyl
0.023750
36. TRI CHEMICAL/COMPOUND ID_0000092524
0.023750
37. CAS#_92-52-4
0.023750
37. CAS#_1313-27-5
0.023750
36. TRI CHEMICAL/COMPOUND ID_0001313275
0.023750
34. CHEMICAL_Molybdenum trioxide
0.023750
63. 6.1 - POTW - TRNS RLSE
0.023353
20. INDUSTRY SECTOR_Electrical Equipment
0.022878
34. CHEMICAL_Barium compounds (except for barium sulfate (CAS No. 7727-43-7))
0.022878
61. 5.5.4 - OTHER DISPOSAL
0.022140
119. 8.9 - PRODUCTION RATIO
0.021989
43. CARCINOGEN_YES
0.021841
```

```
91. OFF-SITE RECYCLED TOTAL
0.021551
113. 8.5 - RECYCLING OFF SIT
0.021548
65. POTW - TOTAL TRANSFERS
0.020846
37. CAS#_110-82-7
0.020707
36. TRI CHEMICAL/COMPOUND ID_0000110827
0.020707
34. CHEMICAL_Cyclohexane
0.020707
34. CHEMICAL_Chlorine dioxide
0.020685
36. TRI CHEMICAL/COMPOUND ID_0010049044
0.020685
37. CAS#_10049-04-4
0.020685
36. TRI CHEMICAL/COMPOUND ID_0007664417
0.020604
37. CAS#_7664-41-7
0.020604
34. CHEMICAL_Ammonia
0.020604
34. CHEMICAL_Nickel compounds
0.020101
104. TOTAL RELEASES
0.019873
64. 6.1 - POTW - TRNS TRT
0.019647
34. CHEMICAL_Manganese  And Manganese Compounds
0.019516
36. TRI CHEMICAL/COMPOUND ID_0000079947
0.019498
37. CAS#_79-94-7
0.019498
34. CHEMICAL_Tetrabromobisphenol A
0.019498
37. CAS#_106-94-5
0.019498
34. CHEMICAL_1-Bromopropane
0.019498
36. TRI CHEMICAL/COMPOUND ID_0000106945
0.019498
37. CAS#_108-88-3
0.019193
36. TRI CHEMICAL/COMPOUND ID_0000108883
0.019193
34. CHEMICAL_Toluene
0.019193
```

```
108. 8.1C - OFF-SITE CONTAIN
0.019191
62. ON-SITE RELEASE TOTAL
0.019068
87. 6.2 - M24
0.018354
34. CHEMICAL_Copper  And Copper Compounds
0.018235
114. 8.6 - TREATMENT ON SITE
0.017919
20. INDUSTRY SECTOR_Chemicals
0.017786
37. CAS#_N458
0.017683
36. TRI CHEMICAL/COMPOUND ID_N458
0.017683
37. CAS#_127-18-4
0.017332
34. CHEMICAL_Tetrachloroethylene
0.017332
36. TRI CHEMICAL/COMPOUND ID_0000127184
0.017332
57. 5.5.2 - LAND TREATMENT
0.017118
36. TRI CHEMICAL/COMPOUND ID_0000137428
0.016879
34. CHEMICAL_Metham sodium
0.016879
37. CAS#_137-42-8
0.016879
34. CHEMICAL_Di(2-ethylhexyl) phthalate
0.016879
34. CHEMICAL_Chromium  and Chromium Compounds(except for chromite ore mined
in the Transvaal Region)                                  0.016879
36. TRI CHEMICAL/COMPOUND ID_0000117817
0.016879
37. CAS#_117-81-7
0.016879
37. CAS#_26471-62-5
0.016879
34. CHEMICAL_Toluene diisocyanate (mixed isomers)
0.016879
36. TRI CHEMICAL/COMPOUND ID_0026471625
0.016879
60. 5.5.3B - OTHER SURFACE I
0.015909
107. 8.1B - ON-SITE OTHER
0.015841
37. CAS#_80-62-6
0.015405
```

```
34. CHEMICAL_Methyl methacrylate
0.015405
36. TRI CHEMICAL/COMPOUND ID_0000080626
0.015405
37. CAS#_71-36-3
0.014794
36. TRI CHEMICAL/COMPOUND ID_0000071363
0.014794
34. CHEMICAL_n-Butyl alcohol
0.014794
34. CHEMICAL_Polycyclic aromatic compounds
0.014785
37. CAS#_N590
0.014785
36. TRI CHEMICAL/COMPOUND ID_N590
0.014785
36. TRI CHEMICAL/COMPOUND ID_N420
0.014631
37. CAS#_N420
0.014631
37. CAS#_95-63-6
0.014523
34. CHEMICAL_1,2,4-Trimethylbenzene
0.014523
36. TRI CHEMICAL/COMPOUND ID_0000095636
0.014523
34. CHEMICAL_Mercury compounds
0.014523
34. CHEMICAL_Cobalt compounds
0.014378
34. CHEMICAL_Lead compounds
0.014001
34. CHEMICAL_Nickel  And Nickel Compounds
0.013776
34. CHEMICAL_Mercury  And Mercury Compounds
0.013776
37. CAS#_55-63-0
0.013776
34. CHEMICAL_Nitroglycerin
0.013776
36. TRI CHEMICAL/COMPOUND ID_0000055630
0.013776
34. CHEMICAL_Dimethylamine
0.013776
36. TRI CHEMICAL/COMPOUND ID_0000124403
0.013776
37. CAS#_124-40-3
0.013776
36. TRI CHEMICAL/COMPOUND ID_0000087865
0.013776
```

```
37. CAS#_87-86-5
0.013776
34. CHEMICAL_Pentachlorophenol
0.013776
34. CHEMICAL_Benzene
0.013700
36. TRI CHEMICAL/COMPOUND ID_0000071432
0.013700
37. CAS#_71-43-2
0.013700
56. 5.5.1B - OTHER LANDFILLS
0.013256
34. CHEMICAL_Manganese compounds
0.013008
36. TRI CHEMICAL/COMPOUND ID_N495
0.012067
37. CAS#_N495
0.012067
35. ELEMENTAL METAL INCLUDED_YES
0.012012
36. TRI CHEMICAL/COMPOUND ID_0001634044
0.011928
37. CAS#_1634-04-4
0.011928
34. CHEMICAL_Methyl tert-butyl ether
0.011928
37. CAS#_137-41-7
0.011928
34. CHEMICAL_Potassium N-methyldithiocarbamate
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000137417
0.011928
34. CHEMICAL_Pyridine
0.011928
37. CAS#_110-86-1
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000110861
0.011928
37. CAS#_25321-14-6
0.011928
36. TRI CHEMICAL/COMPOUND ID_0025321146
0.011928
34. CHEMICAL_Dinitrotoluene (mixed isomers)
0.011928
34. CHEMICAL_Cobalt
0.011928
34. CHEMICAL_Silver
0.011928
37. CAS#_7440-22-4
0.011928
```

```
36. TRI CHEMICAL/COMPOUND ID_0007440484
0.011928
36. TRI CHEMICAL/COMPOUND ID_0007440224
0.011928
37. CAS#_7440-48-4
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000107062
0.011928
37. CAS#_107-06-2
0.011928
34. CHEMICAL_1,2-Dichloroethane
0.011928
34. CHEMICAL_N,N-Dimethylformamide
0.011928
34. CHEMICAL_Acetonitrile
0.011928
34. CHEMICAL_Antimony
0.011928
37. CAS#_68-12-2
0.011928
36. TRI CHEMICAL/COMPOUND ID_0007440360
0.011928
37. CAS#_75-05-8
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000075058
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000068122
0.011928
37. CAS#_7440-36-0
0.011928
37. CAS#_75-09-2
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000075092
0.011928
37. CAS#_67-66-3
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000067663
0.011928
34. CHEMICAL_Chloroform
0.011928
34. CHEMICAL_Dichloromethane
0.011928
37. CAS#_542-75-6
0.011928
34. CHEMICAL_1,3-Dichloropropylene
0.011928
36. TRI CHEMICAL/COMPOUND ID_0000542756
0.011928
37. CAS#_122-39-4
0.011928
```

```
36. TRI CHEMICAL/COMPOUND ID_0000122394
0.011928
34. CHEMICAL_Diphenylamine
0.011928
20. INDUSTRY SECTOR_Textiles
0.011928
37. CAS#_N096
0.010315
36. TRI CHEMICAL/COMPOUND ID_N096
0.010315
34. CHEMICAL_Diethanolamine
0.009838
37. CAS#_111-42-2
0.009838
36. TRI CHEMICAL/COMPOUND ID_0000111422
0.009838
34. CHEMICAL_Lithium carbonate
0.009737
37. CAS#_554-13-2
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000554132
0.009737
34. CHEMICAL_Cyanide compounds
0.009737
34. CHEMICAL_Chloromethane
0.009737
36. TRI CHEMICAL/COMPOUND ID_N106
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000078922
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000074873
0.009737
37. CAS#_N106
0.009737
34. CHEMICAL_sec-Butyl alcohol
0.009737
37. CAS#_78-92-2
0.009737
37. CAS#_74-87-3
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000077736
0.009737
37. CAS#_75-65-0
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000075650
0.009737
37. CAS#_77-73-6
0.009737
34. CHEMICAL_Dicyclopentadiene
0.009737
```

```
34. CHEMICAL_tert-Butyl alcohol
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000123911
0.009737
37. CAS#_123-91-1
0.009737
34. CHEMICAL_1,4-Dioxane
0.009737
34. CHEMICAL_Benzoyl peroxide
0.009737
37. CAS#_94-36-0
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000094360
0.009737
36. TRI CHEMICAL/COMPOUND ID_0000191242
0.009364
34. CHEMICAL_Benzo[g,h,i]perylene
0.009364
37. CAS#_191-24-2
0.009364
37. CAS#_N100
0.009121
36. TRI CHEMICAL/COMPOUND ID_N100
0.009121
36. TRI CHEMICAL/COMPOUND ID_0000100414
0.008448
34. CHEMICAL_Ethylbenzene
0.008448
36. TRI CHEMICAL/COMPOUND ID_N040
0.007994
37. CAS#_N040
0.007994
34. CHEMICAL_n-Hexane
0.007513
37. CAS#_110-54-3
0.007513
36. TRI CHEMICAL/COMPOUND ID_0000110543
0.007513
49. 5.2 - STACK AIR
0.007437
37. CAS#_76-06-2
0.006884
36. TRI CHEMICAL/COMPOUND ID_N078
0.006884
37. CAS#_N078
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000076062
0.006884
37. CAS#_10061-02-6
0.006884
```

```
34. CHEMICAL_trans-1,3-Dichloropropene
0.006884
34. CHEMICAL_Chloropicrin
0.006884
36. TRI CHEMICAL/COMPOUND ID_0010061026
0.006884
34. CHEMICAL_Cadmium compounds
0.006884
34. CHEMICAL_o-Xylene
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000095476
0.006884
37. CAS#_95-47-6
0.006884
37. CAS#_79-06-1
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000079061
0.006884
34. CHEMICAL_Acrylamide
0.006884
37. CAS#_106-42-3
0.006884
34. CHEMICAL_p-Xylene
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000106423
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000108383
0.006884
37. CAS#_79-01-6
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000071556
0.006884
34. CHEMICAL_Trichloroethylene
0.006884
37. CAS#_121-44-8
0.006884
34. CHEMICAL_Triethylamine
0.006884
34. CHEMICAL_m-Xylene
0.006884
37. CAS#_71-55-6
0.006884
37. CAS#_108-38-3
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000079016
0.006884
36. TRI CHEMICAL/COMPOUND ID_0000121448
0.006884
37. CAS#_N020
0.006884
```

```
34. CHEMICAL_Cobalt  And Cobalt Compounds
0.006884
34. CHEMICAL_Arsenic compounds
0.006884
36. TRI CHEMICAL/COMPOUND ID_N020
0.006884
36. TRI CHEMICAL/COMPOUND ID_0007440666
0.006884
37. CAS#_7440-66-6
0.006884
34. CHEMICAL_Zinc (fume or dust)
0.006884
37. CAS#_N450
0.006559
36. TRI CHEMICAL/COMPOUND ID_N450
0.006559
34. CHEMICAL_Lead
0.006307
36. TRI CHEMICAL/COMPOUND ID_0007439921
0.006307
37. CAS#_7439-92-1
0.006307
85. OFF-SITE RELEASE TOTAL
0.006102
37. CAS#_N120
0.005754
34. CHEMICAL_Diisocyanates
0.005754
36. TRI CHEMICAL/COMPOUND ID_N120
0.005754
36. TRI CHEMICAL/COMPOUND ID_0007647010
0.005534
34. CHEMICAL_Hydrochloric acid (acid aerosols including mists, vapors, gas,
fog, and other airborne forms of any particle size)    0.005534
37. CAS#_7647-01-0
0.005534
40. CLASSIFICATION_TRI
0.005124
44. PBT_YES
0.005124
50. 5.3 - WATER
0.004914
34. CHEMICAL_Lead  And Lead Compounds
0.003931
39. CLEAN AIR ACT CHEMICAL_YES
0.003799
88. 6.2 - M26
0.003042
20. INDUSTRY SECTOR_Machinery
0.002529
```

```
109. 8.1D - OFF-SITE OTHER R
0.002275
34. CHEMICAL_Copper compounds
0.001264
34. CHEMICAL_Xylene (mixed isomers)
0.000907
36. TRI CHEMICAL/COMPOUND ID_0001330207
0.000907
37. CAS#_1330-20-7
0.000907
112. 8.4 - RECYCLING ON SITE
0.000898
54. 5.5.1 - LANDFILLS
NaN
69. 6.2 - M40 METAL
NaN
Name: Cancer_Rate_Flag, dtype: float64
```

Defining Our Target/Predictor Variables

```
x = df.loc[ : , df.columns != 'Cancer_Rate_Flag']
y = df["Cancer_Rate_Flag"]
```

Data Processing Techniques

SMOTE

```
#smote usage
sm = SMOTE(random_state=0)
x_sm, y_sm = sm.fit_resample(x, y)

print(f'Over-sampled data: {np.unique(y_sm, return_counts=1)}')
```

```
Over-sampled data: (array([0, 1]), array([2704, 2704]))
```

Feature Selection

```
#featureNames = x_sm.columns
featureNames = x.columns
print(f'Originally, we have {len(featureNames)} features.')

rfc = RandomForestClassifier().fit(x, y)
```

```
Originally, we have 414 features.
```

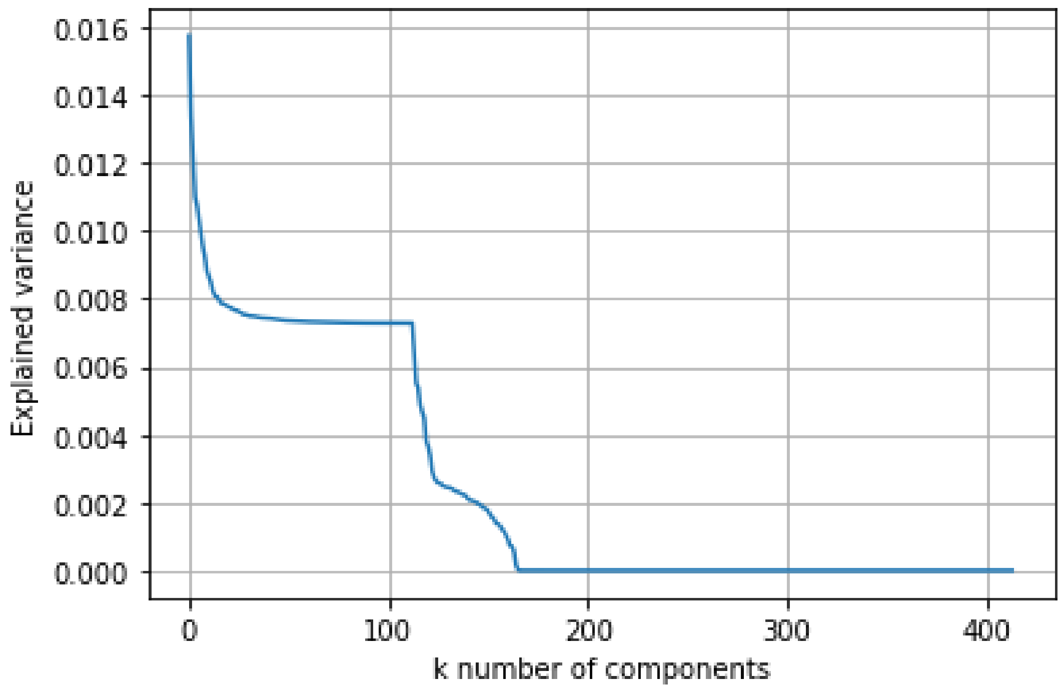
```
#
https://stackoverflow.com/questions/17737300/suppressing-scientific-notation-
in-pandas
pd.options.display.float_format = '{:20,.4f}'.format

def get_feature_importances(cols, importances):
    feats = {}
    for feature, importance in zip(cols, importances):
        feats[feature] = importance
    importances = pd.DataFrame.from_dict(feats,
orient='index').rename(columns={0: 'Gini-importance'})
    importances.sort_values(by='Gini-importance', ascending=False,
inplace=True)

    return importances


importances = get_feature_importances(x_sm.columns, rfc.feature_importances_)
print()
print(importances)

# After Doingo this there is a feature 'Unnamed: 0' which has the highest
Gini-importance of 0.1551.
# It seems to be some sort of ID marker we missed.

importances = rfc.feature_importances_
rfc_importances = pd.DataFrame(data = importances, index=featureNames,
columns=['ImportanceValues'])
# rfc_importances.sort_values(by='ImportanceValues', ascending=False,
inplace=True)

# Sorted Feature Importances | Bar Chart. Way too Small
# plt.barh(rfc_importances.index, rfc_importances.ImportanceValues)

# NOR SORTED. DISCARD?
print("rfc_importances - \n", rfc_importances)

rfc_importances.sort_values(by='ImportanceValues', ascending=False)

selector =
SelectFromModel(estimator=RandomForestClassifier(),threshold=0.001)
X_reduced = selector.fit_transform(x,y)
selector.threshold_
selected_TF = selector.get_support()
print(f'\n** {selected_TF.sum()} features are selected.')


** 122 features are selected.

# Show those selected features.
selected_features = []
for i,j in zip(selected_TF, featureNames):
    if i: selected_features.append(j)
print(f'Selected Features: {selected_features}')
len(selected_features)

Selected Features: ['42. METAL CATEGORY', '48. 5.1 - FUGITIVE AIR', '49. 5.2
- STACK AIR', '50. 5.3 - WATER', '56. 5.5.1B - OTHER LANDFILLS', '57. 5.5.2 -
```

```
LAND TREATMENT', '60. 5.5.3B - OTHER SURFACE I', '61. 5.5.4 - OTHER
DISPOSAL', '62. ON-SITE RELEASE TOTAL', '63. 6.1 - POTW - TRNS RLSE', '64.
6.1 - POTW - TRNS TRT', '65. POTW - TOTAL TRANSFERS', '85. OFF-SITE RELEASE
TOTAL', '87. 6.2 - M24', '88. 6.2 - M26', '91. OFF-SITE RECYCLED TOTAL', '92.
6.2 - M56', '94. OFF-SITE ENERGY RECOVERY T', '96. 6.2 - M50', '101. OFF-SITE
TREATED TOTAL', '103. 6.2 - TOTAL TRANSFER', '104. TOTAL RELEASES', '107.
8.1B - ON-SITE OTHER', '108. 8.1C - OFF-SITE CONTAIN', '109. 8.1D - OFF-SITE
OTHER R', '111. 8.3 - ENERGY RECOVER OF', '112. 8.4 - RECYCLING ON SITE',
'113. 8.5 - RECYCLING OFF SIT', '114. 8.6 - TREATMENT ON SITE', '115. 8.7 -
TREATMENT OFF SITE', '116. PRODUCTION WSTE (8.1-8.7)', '119. 8.9 - PRODUCTION
RATIO', '18. FEDERAL FACILITY_YES', '20. INDUSTRY SECTOR_Chemicals', '20.
INDUSTRY SECTOR_Computers and Electronic Products', '20. INDUSTRY
SECTOR_Fabricated Metals', '20. INDUSTRY SECTOR_Machinery', '20. INDUSTRY
SECTOR_Nonmetallic Mineral Product', '20. INDUSTRY SECTOR_Other', '20.
INDUSTRY SECTOR_Paper', '20. INDUSTRY SECTOR_Petroleum', '20. INDUSTRY
SECTOR_Petroleum Bulk Terminals', '20. INDUSTRY SECTOR_Primary Metals', '20.
INDUSTRY SECTOR_Transportation Equipment', '20. INDUSTRY SECTOR_Wood
Products', '34. CHEMICAL_Acetaldehyde', '34. CHEMICAL_Ammonia', '34.
CHEMICAL_Anthracene', '34. CHEMICAL_Carbon disulfide', '34.
CHEMICAL_Chlorine', '34. CHEMICAL_Diisocyanates', '34. CHEMICAL_Dimethyl
phthalate', '34. CHEMICAL_Formaldehyde', '34. CHEMICAL_Hydrogen cyanide',
'34. CHEMICAL_Hydrogen sulfide', '34. CHEMICAL_Hydroquinone', '34.
CHEMICAL_Lead', '34. CHEMICAL_Lead compounds', '34. CHEMICAL_Methanol', '34.
CHEMICAL_N-Methyl-2-pyrrolidone', '34. CHEMICAL_Nitrate compounds (water
dissociable; reportable only when in aqueous solution)', '34.
CHEMICAL_Phenanthrene', '34. CHEMICAL_Phenol', '34. CHEMICAL_Polycyclic
aromatic compounds', '34. CHEMICAL_Silver', '34. CHEMICAL_Styrene', '34.
CHEMICAL_Sulfuric acid (acid aerosols including mists, vapors, gas, fog, and
other airborne forms of any particle size)', '34. CHEMICAL_Vanadium
compounds', '34. CHEMICAL_Xylene (mixed isomers)', '36. TRI CHEMICAL/COMPOUND
ID_100425', '36. TRI CHEMICAL/COMPOUND ID_106990', '36. TRI CHEMICAL/COMPOUND
ID_108952', '36. TRI CHEMICAL/COMPOUND ID_115071', '36. TRI CHEMICAL/COMPOUND
ID_120127', '36. TRI CHEMICAL/COMPOUND ID_123319', '36. TRI CHEMICAL/COMPOUND
ID_131113', '36. TRI CHEMICAL/COMPOUND ID_1330207', '36. TRI
CHEMICAL/COMPOUND ID_50000', '36. TRI CHEMICAL/COMPOUND ID_67561', '36. TRI
CHEMICAL/COMPOUND ID_7439921', '36. TRI CHEMICAL/COMPOUND ID_7440224', '36.
TRI CHEMICAL/COMPOUND ID_74851', '36. TRI CHEMICAL/COMPOUND ID_74908', '36.
TRI CHEMICAL/COMPOUND ID_75070', '36. TRI CHEMICAL/COMPOUND ID_75150', '36.
TRI CHEMICAL/COMPOUND ID_75456', '36. TRI CHEMICAL/COMPOUND ID_7664417', '36.
TRI CHEMICAL/COMPOUND ID_7783064', '36. TRI CHEMICAL/COMPOUND ID_872504',
'36. TRI CHEMICAL/COMPOUND ID_N120', '36. TRI CHEMICAL/COMPOUND ID_N420',
'36. TRI CHEMICAL/COMPOUND ID_N590', '37. CAS#_100-42-5', '37.
CAS#_10049-04-4', '37. CAS#_108-95-2', '37. CAS#_120-12-7', '37.
CAS#_123-31-9', '37. CAS#_131-11-3', '37. CAS#_1330-20-7', '37.
CAS#_463-58-1', '37. CAS#_50-00-0', '37. CAS#_6/4/7783', '37. CAS#_67-56-1',
'37. CAS#_74-90-8', '37. CAS#_7439-92-1', '37. CAS#_7440-22-4', '37.
CAS#_75-07-0', '37. CAS#_75-15-0', '37. CAS#_7664-41-7', '37.
CAS#_7664-93-9', '37. CAS#_79-21-0', '37. CAS#_872-50-4', '37. CAS#_N120',
'37. CAS#_N420', '37. CAS#_N511', '37. CAS#_N590', '39. CLEAN AIR ACT
```

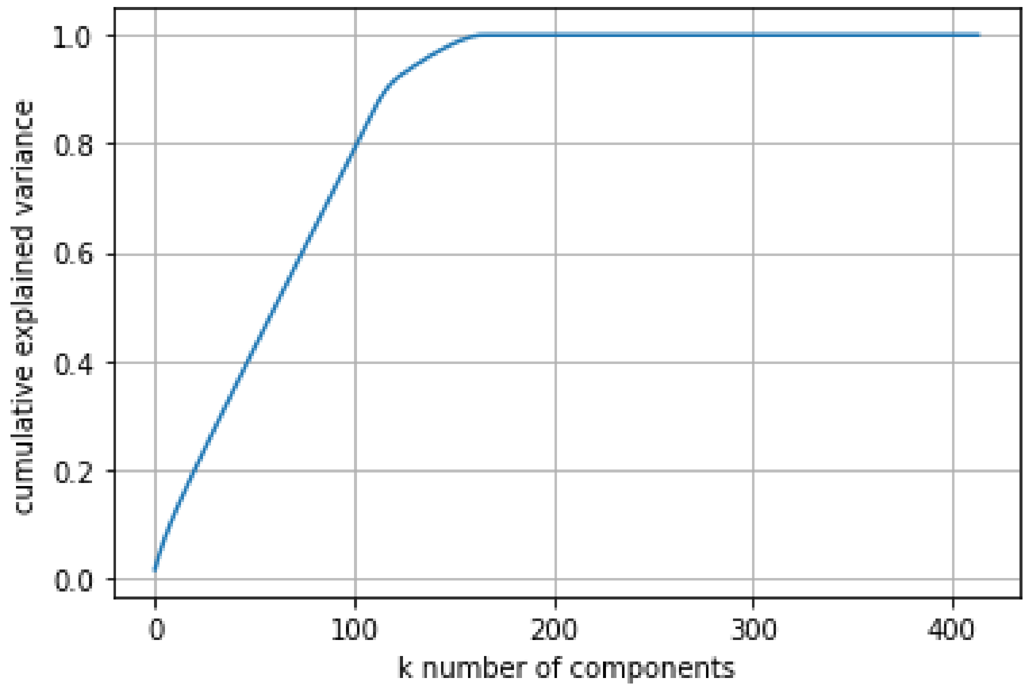
```
CHEMICAL_YES', '40. CLASSIFICATION_TRI', '41. METAL_YES', '43.
CARCINOGEN_YES', '44. PBT_YES', '118. PROD_RATIO_OR_ ACTIVITY_PRODUCTION']

122
```

PCA

```
print(f'\nThe original dataset has {x.shape[1]} features.')

# z_score normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(x)


The original dataset has 414 features.

# Create an instance PCA and build the model using Xn.
# We start from the same number of components as the number of original
# features.
pca_prep = PCA().fit(Xn)
pca_prep.n_components_

414

# Currently, we have 1036 compnents.  We want to find out how many components
# we want to use without losing much information.
pca_prep.explained_variance_
pca_prep.explained_variance_ratio_

#SCREE PLOT
#Find an "elbow" or an inflection point on the Scree plot.
plt.plot(pca_prep.explained_variance_ratio_)
plt.xlabel('k number of components')
plt.ylabel('Explained variance')
plt.grid(True)
plt.show()
```



```python
# Alternative plot using cumulative ratios
plt.plot(np.cumsum(pca_prep.explained_variance_ratio_))
plt.xlabel('k number of components')
plt.ylabel('cumulative explained variance')
plt.grid(True)
plt.show()
```

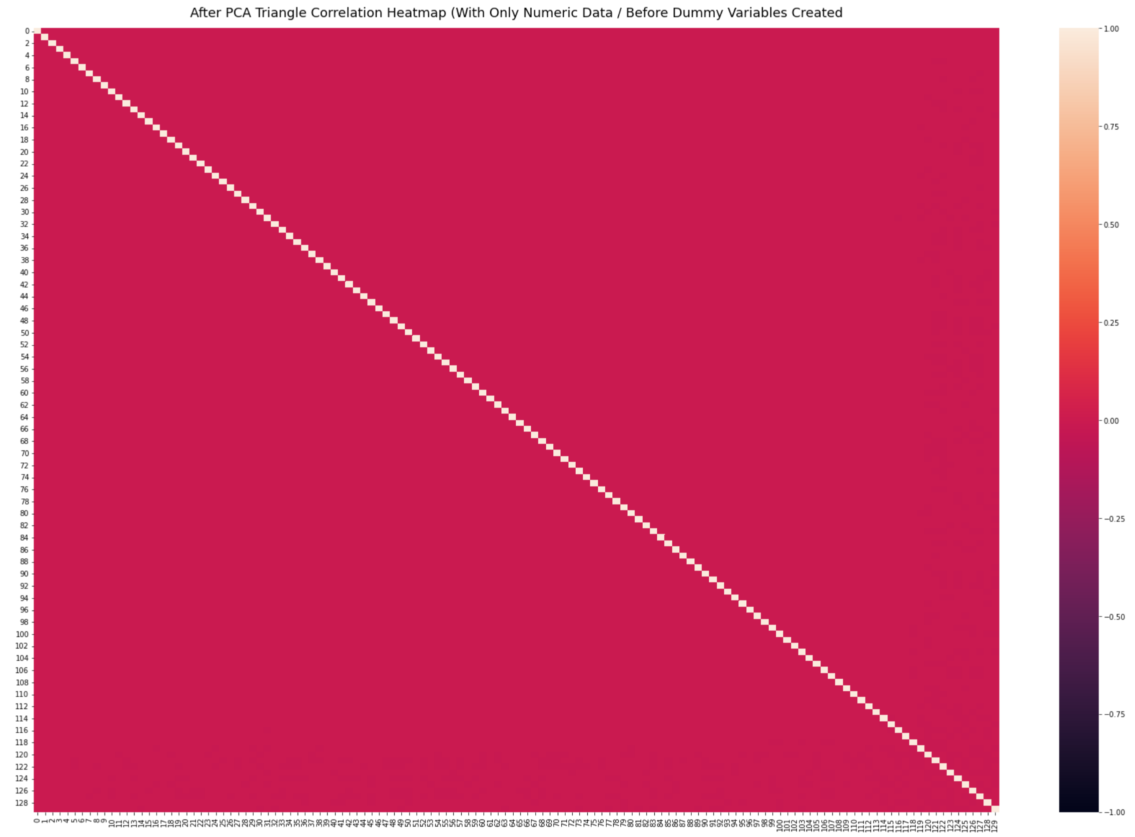


```
# Seeing the scree plot, we choose 130 (DIFF NUMBER)
n_pc = 130
pca = PCA(n_components= n_pc).fit(Xn)

# X_pca has now 410 columns of primary components.
Xp = pca.transform(Xn)
print(f'After PCA, we use {pca.n_components_} components.\n')
```

```
After PCA, we use 130 components.
```

```
cor_mat1  = np.corrcoef(Xp.T)
# eig_vals, eig_vecs = np.linalg.eig(cor_mat1)
# print('Eigenvectors \n%s' %eig_vecs)
# print('\nEigenvalues \n%s' %eig_vals)

post_pca_df = pd.DataFrame(cor_mat1)

#
https://medium.com/@szabo.bibor/how-to-create-a-seaborn-correlation-heatmap-i
n-python-834c0686b88e
plt.figure(figsize=(30, 20))

# define the mask to set the values in the upper triangle to True

heatmap = sns.heatmap(post_pca_df.corr(),
                      vmin=-1, vmax=1
                    #   fmt=".1f", # Round to One Decimal
                    #   annot=True, # Annotate Values
                      #cmap='BrBG', # Change Color Palette,
                    #   linewidth=.5
                      )
heatmap.set_title('After PCA Triangle Correlation Heatmap (With Only Numeric
Data / Before Dummy Variables Created', fontdict={'fontsize':18}, pad=16)
```

```
Text(0.5, 1.0, 'After PCA Triangle Correlation Heatmap (With Only Numeric
Data / Before Dummy Variables Created')
```



```
# plt.figure(figsize=(8, 120))
# heatmap = sns.heatmap(Xp.corr(),
#                                    vmin=-1, vmax=1,
#                                    annot=True,
#                                    linewidth=.1
#                                    #cmap='BrBG'
#                                    )
# heatmap.set_title('After PCA | All Numeric + Dummy Features Correlating w/
Cancer Rate Flag', fontdict={'fontsize':18}, pad=16)

plt.figure(figsize=(8, 120))
heatmap =
sns.heatmap(df.corr()[['Cancer_Rate_Flag']].sort_values(by='Cancer_Rate_Flag'
,

ascending=False),
                                                                  vmin=-1,
vmax=1,
                                                                  annot=True,

linewidth=.1

#cmap='BrBG'
                                                                  )
```

```
heatmap.set_title('After PCA | All Numeric + Dummy Features Correlating w/
Cancer Rate Flag', fontdict={'fontsize':18}, pad=16)

# Split the data into training and testing subsets.

X_train, X_test, y_train, y_test = train_test_split(x,y,test_size =.2,
                                             random_state=1234,stratify=y)

Xp_train, Xp_test, yp_train, yp_test = train_test_split(Xp,y,test_size =.2,
                                             random_state=1234,stratify=y)

# Creating two random forest models: one using the original and the other
using
# the transformed data.

rfcm = RandomForestClassifier().fit(X_train, y_train)
rfcm_p = RandomForestClassifier().fit(Xp_train, yp_train)

y_pred = rfcm.predict(X_test)

y_pred_p = rfcm_p.predict(Xp_test)

# Comparing the performance of each model.
report_original = classification_report(y_test, y_pred)
report_pca = classification_report(yp_test, y_pred_p)
print(f'Classification Report - original\n{report_original}')
print(f'Classification Report - pca\n{report_pca}')

Classification Report - original
              precision    recall  f1-score   support

           0       0.93      0.99      0.96       541
           1       0.77      0.33      0.46        61

    accuracy                           0.92       602
   macro avg       0.85      0.66      0.71       602
weighted avg       0.91      0.92      0.91       602

Classification Report - pca
              precision    recall  f1-score   support

           0       0.92      0.95      0.93       541
           1       0.37      0.25      0.29        61

    accuracy                           0.88       602
   macro avg       0.64      0.60      0.61       602
weighted avg       0.86      0.88      0.87       602
```

Using RandomizedSearchCV

```
# Selection of parameter Values
nnm_r = MLPClassifier()
params = {'hidden_layer_sizes':[(20), (30)],'activation':
          ['logistic', 'tanh','relu'], 'max_iter': [4000,5000]}

start_r = time.time()
rand_src = RandomizedSearchCV(estimator= nnm_r, param_distributions = params,
n_iter=6)
rand_src.fit(Xn,y)
end_r = time.time()
```

Generate a Report

```
# Generate a Report
print('\n\n   **Report**')
print(f'The best estimator: {rand_src.best_estimator_}')
print(f'The best parameters:\n {rand_src.best_params_}')
print(f'The best score: {rand_src.best_score_:.4f}')
print(f'Total run time for RandomizedSearchCV: {(end_r - start_r):.2f}
seconds')
```

```
   **Report**
The best estimator: MLPClassifier(activation='logistic',
hidden_layer_sizes=30, max_iter=5000)
The best parameters:
 {'max_iter': 5000, 'hidden_layer_sizes': 30, 'activation': 'logistic'}
The best score: 0.8856
Total run time for RandomizedSearchCV: 232.78 seconds
```

Grid Search

```
# Grid Search
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import GridSearchCV
import time
nnm = MLPClassifier()
params = {'hidden_layer_sizes':[(20), (30)],
          'activation':['logistic','relu', 'tanh'], 'max_iter': [4000,5000]}
# As for cv (cross validation), cv =5 is a default value.
# We measure the time taken to complete a grid search.
start = time.time()
src = GridSearchCV(estimator= nnm, param_grid= params)
src.fit(Xn, y)
end = time.time()

rf = RandomForestRegressor()
```

```
n_estimators = [5,20,50,100] # number of trees in the random forest
max_features = ['auto', 'sqrt'] # number of features in consideration at
every split
max_depth = [int(x) for x in np.linspace(10, 120, num = 12)] # maximum number
of levels allowed in each decision tree
min_samples_split = [2, 6, 10] # minimum sample number to split a node
min_samples_leaf = [1, 3, 4] # minimum sample number that can be stored in a
leaf node
bootstrap = [True, False] # method used to sample data points

random_grid = {'n_estimators': n_estimators,

'max_features': max_features,

'max_depth': max_depth,

'min_samples_split': min_samples_split,
'min_samples_leaf': min_samples_leaf,

'bootstrap': bootstrap}
from sklearn.model_selection import RandomizedSearchCV

rf_random = RandomizedSearchCV(estimator = rf,param_distributions =
random_grid,
               n_iter = 100, cv = 5, verbose=2, random_state=35, n_jobs = -1)

rf = RandomForestRegressor()

# "Is there a reason this is twice?" - Nick
rf_random = RandomizedSearchCV(estimator = rf,param_distributions =
random_grid,
               n_iter = 100, cv = 5, verbose=2, random_state=35, n_jobs = -1)
rf_random.fit(Xn,y)
```

```
Fitting 5 folds for each of 100 candidates, totalling 500 fits

RandomizedSearchCV(cv=5, estimator=RandomForestRegressor(), n_iter=100,
                   n_jobs=-1,
                   param_distributions={'bootstrap': [True, False],
                                        'max_depth': [10, 20, 30, 40, 50, 60,
                                                      70, 80, 90, 100, 110,
                                                      120],
                                        'max_features': ['auto', 'sqrt'],
                                        'min_samples_leaf': [1, 3, 4],
                                        'min_samples_split': [2, 6, 10],
                                        'n_estimators': [5, 20, 50, 100]},
                   random_state=35, verbose=2)
```

```
print ('Random grid: ', random_grid, '\n')
# print the best parameters
print ('Best Parameters: ', rf_random.best_params_, ' \n')
```

```
Random grid:  {'n_estimators': [5, 20, 50, 100], 'max_features': ['auto',
'sqrt'], 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120],
'min_samples_split': [2, 6, 10], 'min_samples_leaf': [1, 3, 4], 'bootstrap':
[True, False]}

Best Parameters:  {'n_estimators': 20, 'min_samples_split': 2,
'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 70, 'bootstrap':
False}
```

Machine Learing Algorithms

K Fold Technique

```
dtmc = DecisionTreeClassifier()
rfmc = RandomForestClassifier()
gbmc =
GradientBoostingClassifier(max_depth=40,n_estimators=50,learning_rate=.1)

dtmc_mean_score = np.mean(cross_val_score(dtmc,x_sm,y_sm,cv=5))
rfmc_mean_score = np.mean(cross_val_score(rfmc,x_sm,y_sm,cv=5))
gbmc_mean_score = np.mean(cross_val_score(gbmc,x_sm,y_sm,cv=5))

print('**\n Mean Scores (Accuracies) **')
print(f'Mean Score for Decision Tree: {dtmc_mean_score:.4f}')
print(f'Mean Score for Random Forest: {rfmc_mean_score:.4f}')
print(f'Mean Score for Gradient Boosting: {gbmc_mean_score:.4f}')
```

```
**
 Mean Scores (Accuracies) **
Mean Score for Decision Tree: 0.9290
Mean Score for Random Forest: 0.9664
Mean Score for Gradient Boosting: 0.9323
```

Skipped LOG Transform of highly skewed variables

Gradient Boosting

Original Dataset

```
import time

# With Original Dataset
from sklearn.ensemble import GradientBoostingClassifier

#Time
start = time.time()
```

```
# Split the data for training and testing.
X_train, X_test, y_train, y_test = train_test_split(x, y,
   test_size =.3,random_state=1234, stratify=y)

gbc = GradientBoostingClassifier()
gbc.fit(X_train, y_train)

# Prediction
y_pred = gbc.predict(X_test)

# Calculate the accuracy
gbc.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
print('Gradient Boosting')
print(metrics.classification_report(y_test,y_pred))

#Time
end = time.time()

print(f'Total run time for GradientBoosting with Original Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gbc,x,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Gradient Boosting: {mean_score:.4f}')
```

```
Accuracy: 0.95
Gradient Boosting
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       655
           1       0.89      0.55      0.68        76

    accuracy                           0.95       731
   macro avg       0.92      0.77      0.83       731
weighted avg       0.94      0.95      0.94       731

Total run time for GradientBoosting with Original Dataset: 1.18 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Gradient Boosting: 0.8843
```

<u>SMOTE Applied Dataset</u>

```
# SMOTE applied dataset

#Time
```

```
start = time.time()

# Split the data for training and testing.
X_sm_train, X_sm_test, y_sm_train, y_sm_test = \
   train_test_split(x_sm,y_sm,random_state=1234,stratify=y_sm)

gbc = GradientBoostingClassifier()
gbc.fit(X_sm_train, y_sm_train)

# Prediction
y_pred_sm = gbc.predict(X_test)

# Calculate the accuracy
gbc.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
print('''*** \nConfusion Matrix and Classification Report
      using the Classifier built from the Oversampled Data ***''')
cm_sm = metrics.confusion_matrix(y_test,y_pred_sm)
print('\n',cm_sm,'\n')
print(metrics.classification_report(y_test,y_pred_sm))

#Time
end = time.time()

print(f'Total run time for GradientBoosting with Smote Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gbc,x_sm, y_sm,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Gradient Boosting: {mean_score:.4f}')
```

```
Accuracy: 0.95
***
Confusion Matrix and Classification Report
      using the Classifier built from the Oversampled Data ***

 [[613  42]
 [ 13  63]]

              precision    recall  f1-score   support

           0       0.98      0.94      0.96       655
           1       0.60      0.83      0.70        76

    accuracy                           0.92       731
   macro avg       0.79      0.88      0.83       731
```

```
weighted avg       0.94      0.92      0.93       731

Total run time for GradientBoosting with Smote Dataset: 2.30 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Gradient Boosting: 0.9197
```

Feature Selection Dataset

```
# Feature Selection Dataset

#Time
start = time.time()

# Split the data for training and testing.
X_train, X_test, y_train, y_test = train_test_split(X_reduced, y,
   test_size =.3,random_state=1234, stratify=y)

gbc = GradientBoostingClassifier()
gbc.fit(X_train, y_train)

# Prediction
y_pred = gbc.predict(X_test)

# Build a confusion matrix and calculate evaluation ratios
print('Gradient Boosting')
print(metrics.classification_report(y_test,y_pred))

#Time
end = time.time()

print(f'Total run time for GradientBoosting with Feature Selection Dataset:
{(end - start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gbc,X_reduced, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Gradient Boosting: {mean_score:.4f}')
```

```
Gradient Boosting
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       655
           1       0.90      0.57      0.69        76

    accuracy                           0.95       731
   macro avg       0.92      0.78      0.83       731
weighted avg       0.95      0.95      0.94       731

Total run time for GradientBoosting with Feature Selection Dataset: 0.67
```

```
seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Gradient Boosting: 0.8835
```

PCA Applied Dataset

```
#PCA Applied Dataset

#Time
start = time.time()

# Split the data for training and testing.
X_train, X_test, y_train, y_test = train_test_split(Xp, y,
   test_size =.3,random_state=1234, stratify=y)

gbc = GradientBoostingClassifier()
gbc.fit(X_train, y_train)

# Prediction
y_pred = gbc.predict(X_test)

# Calculate the accuracy
gbc.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
print('Gradient Boosting')
print(metrics.classification_report(y_test,y_pred))

#Time
end = time.time()

print(f'Total run time for GradientBoosting with PCA Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gbc,Xp, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Gradient Boosting: {mean_score:.4f}')
```

```
Accuracy: 0.89
Gradient Boosting
              precision    recall  f1-score   support

           0       0.91      0.98      0.94       655
           1       0.43      0.16      0.23        76

    accuracy                           0.89       731
   macro avg       0.67      0.57      0.59       731
```

```
weighted avg       0.86      0.89      0.87       731

Total run time for GradientBoosting with PCA Dataset: 6.06 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Gradient Boosting: 0.8789
```

Neural Network w/ GridSearchCV

Original Dataset

```
# With Original Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(x)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nmm = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nmm.fit(X_train, y_train)

## predict test set
y_pred = nmm.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes': [(20),(30)],
          'activation':['logistic','relu','tanh'],
          'max_iter':[4000, 5000]}

start = time.time()

src = GridSearchCV(estimator=nnm, param_grid=params)
src.fit(Xn, y)

end = time.time()

# generating report
print('Neural network report:')
print(f'The best estimator is : {src.best_estimator_}')
print(f'The best parameters are : {src.best_params_}')
print(f'The best score is : {src.best_score_:.4f}')
print(f'Total run time for GridSearchCV is : {(end - start):.2f}seconds')

results = pd.DataFrame(src.cv_results_)
```

```
[[769  43]
 [ 68  23]]
Accuracy:  0.8770764119601329
Precision:  0.3484848484848485
Recall:  0.25274725274725274
F1 score:  0.2929936305732484
Neural network report:
The best estimator is : MLPClassifier(activation='logistic',
hidden_layer_sizes=20, max_iter=4000)
The best parameters are : {'activation': 'logistic', 'hidden_layer_sizes':
20, 'max_iter': 4000}
The best score is : 0.8853
Total run time for GridSearchCV is : 347.67seconds
```

<u>SMOTE Applied Dataset</u>

```
# SMOTE applied dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(x_sm)

X_train, X_test, y_train, y_test = train_test_split(Xn, y_sm, test_size
=.3,random_state=1234, stratify=y_sm)

# selection of parameter values
nmm = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nmm.fit(X_train, y_train)

## predict test set
y_pred = nmm.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes': [(20),(30)],
          'activation':['logistic','relu','tanh'],
          'max_iter':[4000, 5000]}

start = time.time()

src = GridSearchCV(estimator=nmm, param_grid=params)
```

```
src.fit(Xn, y_sm)

end = time.time()

# generating report
print('Neural network report:')
print(f'The best estimator is : {src.best_estimator_}')
print(f'The best parameters are : {src.best_params_}')
print(f'The best score is : {src.best_score_:.4f}')
print(f'Total run time for GridSearchCV is : {(end - start):.2f}seconds')

results = pd.DataFrame(src.cv_results_)
```

Feature Selection Dataset

```
# Feature Selection Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(X_reduced)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nmm = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nmm.fit(X_train, y_train)

## predict test set
y_pred = nmm.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes': [(20),(30)],
          'activation':['logistic','relu','tanh'],
          'max_iter':[4000, 5000]}

start = time.time()

src = GridSearchCV(estimator=nnm, param_grid=params)
src.fit(Xn, y)

end = time.time()
```

```
# generating report
print('Neural network report:')
print(f'The best estimator is : {src.best_estimator_}')
print(f'The best parameters are : {src.best_params_}')
print(f'The best score is : {src.best_score_:.4f}')
print(f'Total run time for GridSearchCV is : {(end - start):.2f}seconds')

results = pd.DataFrame(src.cv_results_)
```

```
[[798  14]
 [ 78  13]]
Accuracy:  0.8981173864894795
Precision:  0.48148148148148145
Recall:  0.14285714285714285
F1 score:  0.22033898305084745
Neural network report:
The best estimator is : MLPClassifier(hidden_layer_sizes=30, max_iter=5000)
The best parameters are : {'activation': 'relu', 'hidden_layer_sizes': 30,
'max_iter': 5000}
The best score is : 0.8906
Total run time for GridSearchCV is : 283.97seconds
```

PCA Applied Dataset

```
#PCA Applied Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(Xp)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nmm = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nmm.fit(X_train, y_train)

## predict test set
y_pred = nmm.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))


params = {'hidden_layer_sizes': [(20),(30)],
          'activation':['logistic','relu','tanh'],
          'max_iter':[4000, 5000]}

start = time.time()

src = GridSearchCV(estimator=nnm, param_grid=params)
src.fit(Xn, y)

end = time.time()

# generating report
print('Neural network report:')
print(f'The best estimator is : {src.best_estimator_}')
print(f'The best parameters are : {src.best_params_}')
print(f'The best score is : {src.best_score_:.4f}')
print(f'Total run time for GridSearchCV is : {(end - start):.2f}seconds')

results = pd.DataFrame(src.cv_results_)
```

```
[[779  33]
 [ 73  18]]
Accuracy:  0.8826135105204873
Precision:  0.35294117647058826
Recall:  0.1978021978021978
F1 score:  0.2535211267605634
```

Neural Network w/ RandomizedSearchCV

Original Dataset

```
# With Original Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(x)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nnm_r = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nnm_r.fit(X_train, y_train)

## predict test set
y_pred = nnm_r.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

# selection of parameter values
params = {'hidden_layer_sizes':[(20), (30)],
          'activation':['logistic', 'tanh','relu'],
          'max_iter': [4000,5000]}

start_r = time.time()
rand_src = RandomizedSearchCV(estimator= nnm_r, param_distributions = params,
n_iter=6)
rand_src.fit(Xn,y)
end_r = time.time()

# generate a Report
print('Neural network report:')
print(f'The best estimator is : {rand_src.best_estimator_}')
print(f'The best parameters are :\n {rand_src.best_params_}')
print(f'The best score is : {rand_src.best_score_:.4f}')
print(f'Total run time for RandomizedSearchCV is : {(end_r - start_r):.2f}
seconds')

results_rgs = pd.DataFrame(rand_src.cv_results_)
```

SMOTE Applied Dataset

```
# SMOTE applied dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(x_sm)

X_train, X_test, y_train, y_test = train_test_split(Xn, y_sm, test_size
=.3,random_state=1234, stratify=y_sm)

# selection of parameter values
nnm_r = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nnm_r.fit(X_train, y_train)

## predict test set
y_pred = nnm_r.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes':[(20), (30)],
          'activation':['logistic', 'tanh','relu'],
          'max_iter': [4000,5000]}

start_r = time.time()
rand_src = RandomizedSearchCV(estimator= nnm_r, param_distributions = params,
n_iter=6)
rand_src.fit(Xn,y_sm)
end_r = time.time()

# generate a Report
print('Neural network report:')
print(f'The best estimator is : {rand_src.best_estimator_}')
print(f'The best parameters are :\n {rand_src.best_params_}')
print(f'The best score is : {rand_src.best_score_:.4f}')
print(f'Total run time for RandomizedSearchCV is : {(end_r - start_r):.2f}
seconds')

results_rgs = pd.DataFrame(rand_src.cv_results_)
```

Feature Selection Dataset

```
# Feature Selection Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(X_reduced)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nnm_r = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nnm_r.fit(X_train, y_train)

## predict test set
y_pred = nnm_r.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes':[(20), (30)],
          'activation':['logistic', 'tanh','relu'],
          'max_iter': [4000,5000]}

start_r = time.time()
```

```
rand_src = RandomizedSearchCV(estimator= nnm_r, param_distributions = params,
n_iter=6)
rand_src.fit(Xn,y)
end_r = time.time()

# generate a Report
print('Neural network report:')
print(f'The best estimator is : {rand_src.best_estimator_}')
print(f'The best parameters are :\n {rand_src.best_params_}')
print(f'The best score is : {rand_src.best_score_:.4f}')
print(f'Total run time for RandomizedSearchCV is : {(end_r - start_r):.2f}
seconds')

results_rgs = pd.DataFrame(rand_src.cv_results_)
```

PCA Applied Dataset

```
#PCA Applied Dataset

# normalize the data
scaler = StandardScaler()
Xn = scaler.fit_transform(Xp)

X_train, X_test, y_train, y_test = train_test_split(Xn, y, test_size
=.3,random_state=1234, stratify=y)

# selection of parameter values
nnm_r = MLPClassifier(hidden_layer_sizes=(3), max_iter=1000)
nnm_r.fit(X_train, y_train)

## predict test set
y_pred = nnm_r.predict(X_test)

## confusion matrix
print(metrics.confusion_matrix(y_test, y_pred))
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
print("Precision: ", metrics.precision_score(y_test, y_pred))
print("Recall: ", metrics.recall_score(y_test, y_pred))
print("F1 score: ", metrics.f1_score(y_test, y_pred))

params = {'hidden_layer_sizes':[(20), (30)],
          'activation':['logistic', 'tanh','relu'],
          'max_iter': [4000,5000]}

start_r = time.time()
rand_src = RandomizedSearchCV(estimator= nnm_r, param_distributions = params,
n_iter=6)
rand_src.fit(Xn,y)
end_r = time.time()
```

```
# generate a Report
print('Neural network report:')
print(f'The best estimator is : {rand_src.best_estimator_}')
print(f'The best parameters are :\n {rand_src.best_params_}')
print(f'The best score is : {rand_src.best_score_:.4f}')
print(f'Total run time for RandomizedSearchCV is : {(end_r - start_r):.2f}
seconds')

results_rgs = pd.DataFrame(rand_src.cv_results_)
```

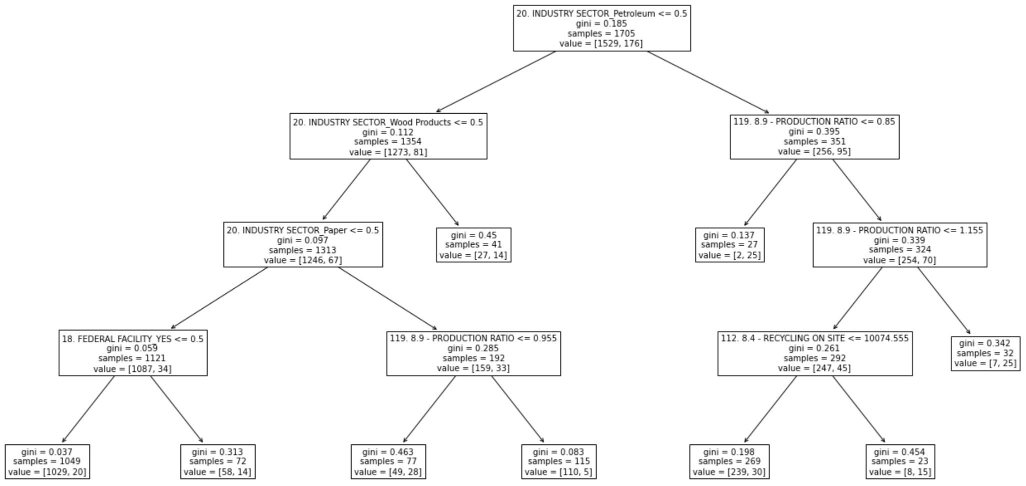
Decision Tree

Original Dataset

```
# Original Dataset
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics, datasets,tree
import matplotlib.pyplot as plt

# %matplotlib inline

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size
=.3,random_state=1234, stratify=y)

# Create a model (object) instantiated from the DecisionTreeClassifier class
dtm = DecisionTreeClassifier(max_depth=4,
    min_samples_split=50,
    min_samples_leaf=20)
# Build a decision tree
dtm.fit(X_train, y_train)

end = time.time()

print(f'Total run time for Decision Tree algorithm with Original Dataset:
{(end - start):.2f} seconds')

# Make predictions using X_test
y_pred = dtm.predict(X_test);
#display(y_pred)
# Calculate the accuracy
dtm.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
```

```
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))
# # Create a decision tree plot.
#fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (8,8), dpi=300)
#tree.plot_tree(dtm,feature_names = featureNames)

plt.figure(figsize=(24,12))
tree.plot_tree(dtm, feature_names=featureNames, fontsize=10)
plt.show()

#k-fold
dtmc_mean_score = np.mean(cross_val_score(dtm,x,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Decision Tree: {dtmc_mean_score:.4f}')
```

```
Total run time for Decision Tree algorithm with Original Dataset: 0.03
seconds
Accuracy: 0.93
              precision    recall  f1-score   support

           0       0.94      0.99      0.96       655
           1       0.83      0.45      0.58        76

    accuracy                           0.93       731
   macro avg       0.88      0.72      0.77       731
weighted avg       0.93      0.93      0.92       731
```



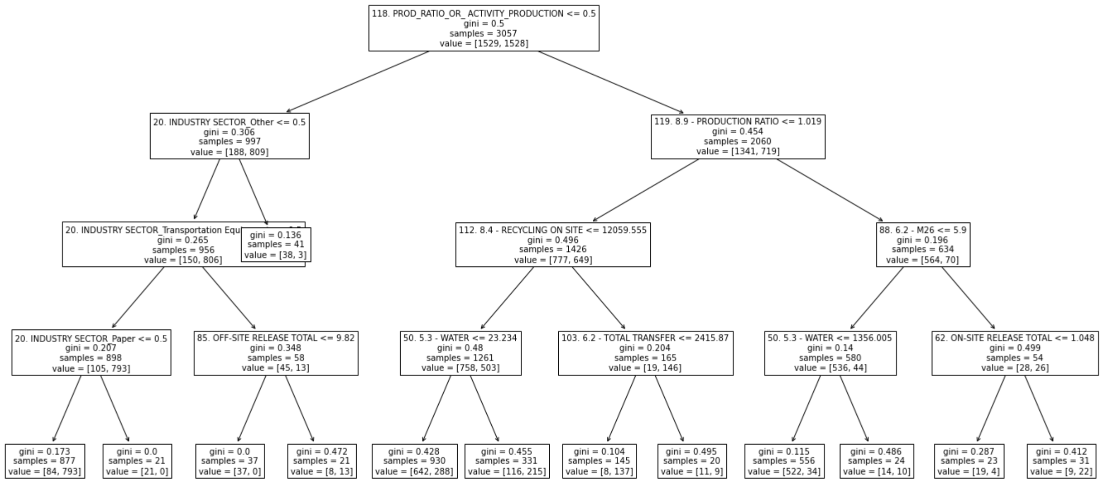
```
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Decision Tree: 0.8982
```

SMOTE Applied Dataset

```
# SMOTE applied Dataset
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics, datasets,tree
import matplotlib.pyplot as plt

# %matplotlib inline

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(x_sm, y_sm, test_size
=.3,random_state=1234, stratify=y_sm)

# Create a model (object) instantiated from the DecisionTreeClassifier class
dtm = DecisionTreeClassifier(max_depth=4,
    min_samples_split=50,
    min_samples_leaf=20)
# Build a decision tree
dtm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for Decision Tree algorithm with SMOTE Dataset: {(end
- start):.2f} seconds')

# Make predictions using X_test
y_pred = dtm.predict(X_test);
#display(y_pred)
# Calculate the accuracy
dtm.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))
# # Create a decision tree plot.
#fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (8,8), dpi=300)
#tree.plot_tree(dtm,feature_names = featureNames)

plt.figure(figsize=(24,12))
tree.plot_tree(dtm, feature_names=featureNames, fontsize=10)
plt.show()


#k-fold
dtmc_mean_score = np.mean(cross_val_score(dtm,x_sm, y_sm,cv=5))
```

```
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Decision Tree: {dtmc_mean_score:.4f}')
```

```
Total run time for Decision Tree algorithm with SMOTE Dataset: 0.05 seconds
Accuracy: 0.81
              precision    recall  f1-score   support

           0       0.78      0.86      0.82       655
           1       0.84      0.76      0.80       656

    accuracy                           0.81      1311
   macro avg       0.81      0.81      0.81      1311
weighted avg       0.81      0.81      0.81      1311
```



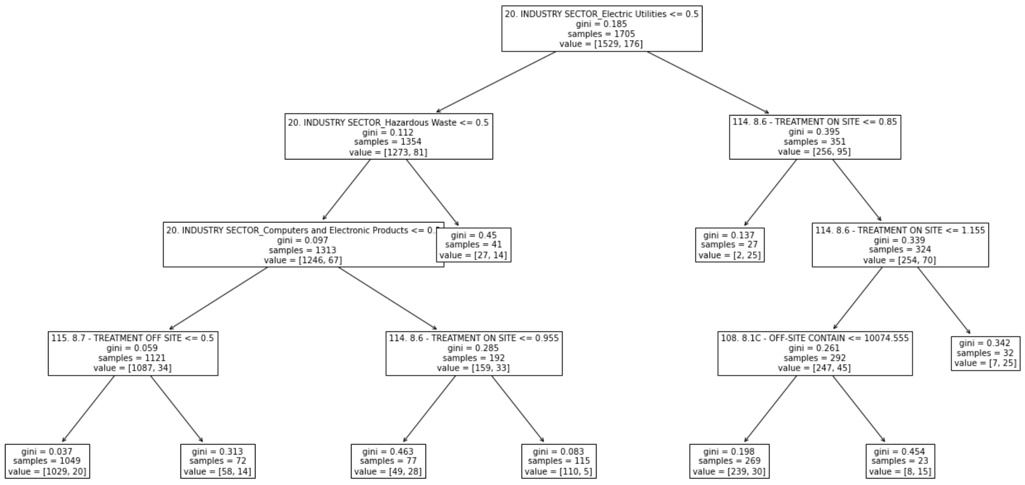
```
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Decision Tree: 0.7942
```

Feature Selection Applied Dataset

```
# Feature Selection applied Dataset
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics, datasets,tree
import matplotlib.pyplot as plt

# %matplotlib inline

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(X_reduced, y, test_size
=.3,random_state=1234, stratify=y)
```

```
# Create a model (object) instantiated from the DecisionTreeClassifier class
dtm = DecisionTreeClassifier(max_depth=4,
    min_samples_split=50,
    min_samples_leaf=20)
# Build a decision tree
dtm.fit(X_train, y_train)

end = time.time()

print(f'Total run time for Decision Tree algorithm with Feature Selection
applied Dataset: {(end - start):.2f} seconds')

# Make predictions using X_test
y_pred = dtm.predict(X_test);
#display(y_pred)
# Calculate the accuracy
dtm.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))
# # Create a decision tree plot.
#fig, axes = plt.subplots(nrows = 1,ncols = 1,figsize = (8,8), dpi=300)
#tree.plot_tree(dtm,feature_names = featureNames)

plt.figure(figsize=(24,12))
tree.plot_tree(dtm, feature_names=featureNames, fontsize=10)
plt.show()


#k-fold
dtmc_mean_score = np.mean(cross_val_score(dtm,X_reduced, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Decision Tree: {dtmc_mean_score:.4f}')
```

```
Total run time for Decision Tree algorithm with Feature Selection applied
Dataset: 0.02 seconds
Accuracy: 0.93
              precision    recall  f1-score   support

           0       0.94      0.99      0.96       655
           1       0.83      0.45      0.58        76

    accuracy                           0.93       731
   macro avg       0.88      0.72      0.77       731
weighted avg       0.93      0.93      0.92       731
```



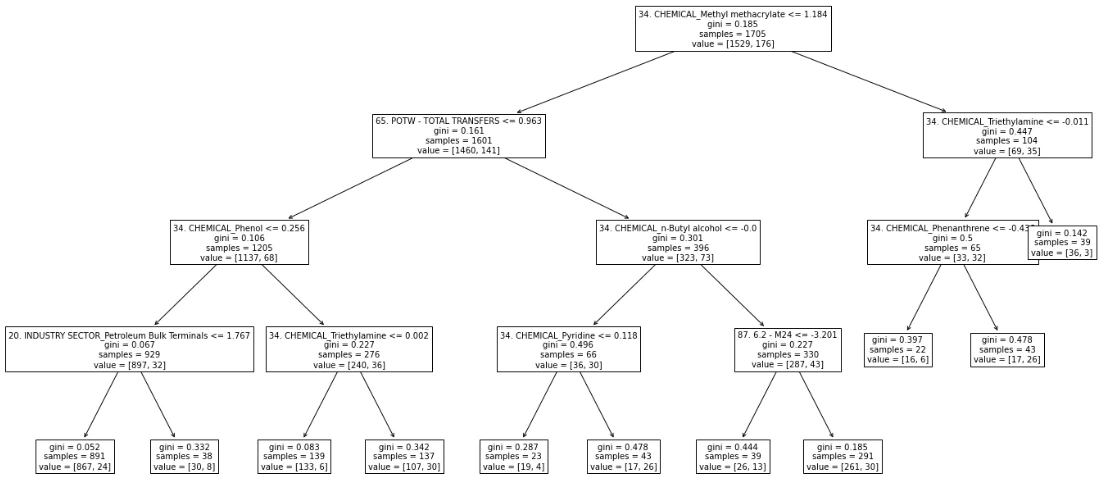
```
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Decision Tree: 0.8982
```

PCA Applied Dataset

```python
# PCA applied Dataset
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics, datasets,tree
import matplotlib.pyplot as plt

# %matplotlib inline

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(Xp, y, test_size
=.3,random_state=1234, stratify=y)

# Create a model (object) instantiated from the DecisionTreeClassifier class
dtm = DecisionTreeClassifier(max_depth=4,
    min_samples_split=50,
    min_samples_leaf=20)
# Build a decision tree
dtm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for DEcision Tree algorithm with PCA applied Dataset:
{(end - start):.2f} seconds')


# Make predictions using X_test
```

```
y_pred = dtm.predict(X_test);
#display(y_pred)
# Calculate the accuracy
dtm.score(X_test,y_test)
print(f'Accuracy: {metrics.accuracy_score(y_test, y_pred):.2f}')

# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))

# # Create a decision tree plot.
#fig, axes = plt.subplots(nrows=1, ncols=1, figsize=(24, 18), dpi=200)
#tree.plot_tree(dtm, ax=axes, fontsize=25)
#tree.plot_tree(dtm, feature_names=featureNames)

plt.figure(figsize=(24,12))
tree.plot_tree(dtm, feature_names=featureNames, fontsize=10)
plt.show()

#k-fold
dtmc_mean_score = np.mean(cross_val_score(dtm,Xp, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Decision Tree: {dtmc_mean_score:.4f}')
```

```
Total run time for DEcision Tree algorithm with PCA applied Dataset: 0.09
seconds
Accuracy: 0.89
              precision    recall  f1-score   support

           0       0.91      0.98      0.94       655
           1       0.45      0.17      0.25        76

    accuracy                           0.89       731
   macro avg       0.68      0.57      0.59       731
weighted avg       0.86      0.89      0.87       731
```



```
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Decision Tree: 0.8806
```

Random Forest Classifier

Original Dataset

```python
#Original Dataset
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics, datasets
# import matplotlib.pyplot as plt

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(x,y,test_size
=.3,random_state=1234, stratify=y)

# Create a model (object) for classification
rfcm = RandomForestClassifier()
# Build a random forest classification model
rfcm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for Random Forest algorithm with Original Dataset:
{(end - start):.2f} seconds')

# Make predictions using the test data
y_pred = rfcm.predict(X_test)
# Print the performance scores.
print('\n ** Performance Scores **')
# Calculate accuracy
```

```
accuracy = rfcm.score(X_test, y_test)
print(f'Accuray: {accuracy:.2f}')
# print('Accuracy: {0:.2f}'.format(accuracy))
# Build a confusion matrix and show the Classification Report
cm = metrics.confusion_matrix(y_test,y_pred)
print('\nConfusion Matrix\n',cm)
print('\nClassification Report\n')
print(metrics.classification_report(y_test,y_pred))

#k-fold
rfmc_mean_score = np.mean(cross_val_score(rfcm,x,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Random Forest: {rfmc_mean_score:.4f}')
```

```
Total run time for Random Forest algorithm with Original Dataset: 0.41
seconds

 ** Performance Scores **
Accuray: 0.92

Confusion Matrix
 [[645  10]
 [ 45  31]]

Classification Report

              precision    recall  f1-score   support

           0       0.93      0.98      0.96       655
           1       0.76      0.41      0.53        76

    accuracy                           0.92       731
   macro avg       0.85      0.70      0.74       731
weighted avg       0.92      0.92      0.91       731

**Mean Score (Accuracy) after applying k-fold**
Mean Score for Random Forest: 0.8966
```

SMOTE Applied Dataset

```
#SMOTE applied Dataset
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics, datasets
# import matplotlib.pyplot as plt

#Time
start = time.time()
```

```
X_train, X_test, y_train, y_test = train_test_split(x_sm,y_sm,test_size
=.3,random_state=1234, stratify=y_sm)

# Create a model (object) for classification
rfcm = RandomForestClassifier()
# Build a random forest classification model
rfcm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for Random Forest algorithm with SMOTE applied
Dataset: {(end - start):.2f} seconds')

# Make predictions using the test data
y_pred = rfcm.predict(X_test)
# Print the performance scores.
print('\n ** Performance Scores **')
# Calculate accuracy
accuracy = rfcm.score(X_test, y_test)
print(f'Accuray: {accuracy:.2f}')
# print('Accuracy: {0:.2f}'.format(accuracy))
# Build a confusion matrix and show the Classification Report
cm = metrics.confusion_matrix(y_test,y_pred)
print('\nConfusion Matrix\n',cm)
print('\nClassification Report\n')
print(metrics.classification_report(y_test,y_pred))

#k-fold
rfmc_mean_score = np.mean(cross_val_score(rfcm,x_sm,y_sm,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Random Forest: {rfmc_mean_score:.4f}')
```

```
Total run time for Random Forest algorithm with SMOTE applied Dataset: 0.63
seconds

 ** Performance Scores **
Accuray: 0.97

Confusion Matrix
 [[627  28]
 [ 16 640]]

Classification Report

              precision    recall  f1-score   support

           0       0.98      0.96      0.97       655
           1       0.96      0.98      0.97       656

    accuracy                           0.97      1311
```

```
   macro avg       0.97      0.97      0.97      1311
weighted avg       0.97      0.97      0.97      1311

**Mean Score (Accuracy) after applying k-fold**
Mean Score for Random Forest: 0.9650
```

Feature Selection Applied Dataset

```
#Feature Selection applied Dataset
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics, datasets
# import matplotlib.pyplot as plt

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(X_reduced,y,test_size
=.3,random_state=1234, stratify=y)

# Create a model (object) for classification
rfcm = RandomForestClassifier()
# Build a random forest classification model
rfcm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for Decision Tree algorithm with Feature Selection
applied Dataset: {(end - start):.2f} seconds')

# Make predictions using the test data
y_pred = rfcm.predict(X_test)
# Print the performance scores.
print('\n ** Performance Scores **')
# Calculate accuracy
accuracy = rfcm.score(X_test, y_test)
print(f'Accuray: {accuracy:.2f}')
# print('Accuracy: {0:.2f}'.format(accuracy))
# Build a confusion matrix and show the Classification Report
cm = metrics.confusion_matrix(y_test,y_pred)
print('\nConfusion Matrix\n',cm)
print('\nClassification Report\n')
print(metrics.classification_report(y_test,y_pred))

#k-fold
rfmc_mean_score = np.mean(cross_val_score(rfcm,X_reduced,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Random Forest: {rfmc_mean_score:.4f}')
```

```
Total run time for Decision Tree algorithm with Feature Selection applied
Dataset: 0.28 seconds
```

```
 ** Performance Scores **
Accuray: 0.93

Confusion Matrix
 [[647   8]
 [ 44  32]]

Classification Report

              precision    recall  f1-score   support

           0       0.94      0.99      0.96       655
           1       0.80      0.42      0.55        76

    accuracy                           0.93       731
   macro avg       0.87      0.70      0.76       731
weighted avg       0.92      0.93      0.92       731

**Mean Score (Accuracy) after applying k-fold**
Mean Score for Random Forest: 0.9031
```

<u>PCA Applied Dataset</u>

```
#PCA applied Dataset
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics, datasets
# import matplotlib.pyplot as plt

#Time
start = time.time()

X_train, X_test, y_train, y_test = train_test_split(Xp,y,test_size
=.3,random_state=1234, stratify=y)

# Create a model (object) for classification
rfcm = RandomForestClassifier()
# Build a random forest classification model
rfcm.fit(X_train, y_train)

end = time.time()
print(f'Total run time for Decision Tree algorithm with PCA applied Dataset:
{(end - start):.2f} seconds')


# Make predictions using the test data
y_pred = rfcm.predict(X_test)
# Print the performance scores.
```

```
print('\n ** Performance Scores **')
# Calculate accuracy
accuracy = rfcm.score(X_test, y_test)
print(f'Accuray: {accuracy:.2f}')
# print('Accuracy: {0:.2f}'.format(accuracy))
# Build a confusion matrix and show the Classification Report
cm = metrics.confusion_matrix(y_test,y_pred)
print('\nConfusion Matrix\n',cm)
print('\nClassification Report\n')
print(metrics.classification_report(y_test,y_pred))

#k-fold
rfmc_mean_score = np.mean(cross_val_score(rfcm,Xp,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Random Forest: {rfmc_mean_score:.4f}')
```

```
Total run time for Decision Tree algorithm with PCA applied Dataset: 1.08
seconds

 ** Performance Scores **
Accuray: 0.89

Confusion Matrix
 [[637  18]
 [ 60  16]]

Classification Report

              precision    recall  f1-score   support

           0       0.91      0.97      0.94       655
           1       0.47      0.21      0.29        76

    accuracy                           0.89       731
   macro avg       0.69      0.59      0.62       731
weighted avg       0.87      0.89      0.87       731

**Mean Score (Accuracy) after applying k-fold**
Mean Score for Random Forest: 0.8740
```

Naive Bayesian Classification

Original Dataset

```
#NB with Original Dataset
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics, datasets

#Time
```

```
start = time.time()

#Split Dataset
X_train, X_test, y_train, y_test = train_test_split(x,y,test_size =.3,
random_state=1234, stratify=y)
# Create a Bayesian Classifier instance for classification
gnb = GaussianNB()
# Build a Bayesian Classification Model and predict the type using the test
data.
gnb.fit(X_train, y_train)
# Calculate the posteriori probabilities
p = gnb.predict_proba(X_test)
# Predict the target value using the test data.
y_pred = gnb.predict(X_test)

#Time
end = time.time()

# Calculate the accuracy
accuracy = gnb.score(X_test, y_test)
# This is the old formatting method.
# print('Accuracy: {:.4f}'.format(accuracy))
print(f'Accuracy: {accuracy: .4f}')
# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))
print(f'Total run time for GradientBoosting with Original Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gnb,x,y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Naive Bayesian: {mean_score:.4f}')
```

```
Accuracy:  0.2449
              precision    recall  f1-score   support

           0       0.97      0.16      0.28       655
           1       0.12      0.96      0.21        76

    accuracy                           0.24       731
   macro avg       0.54      0.56      0.24       731
weighted avg       0.88      0.24      0.27       731

Total run time for GradientBoosting with Original Dataset: 0.05 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Naive Bayesian: 0.2603
```

SMOTE Applied Dataset

```
#NB with Smote Dataset
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics, datasets

#Time
start = time.time()

#Split Dataset
X_sm_train, X_sm_test, y_sm_train, y_sm_test = \
   train_test_split(x_sm,y_sm,random_state=1234,stratify=y_sm)
# Create a Bayesian Classifier instance for classification
gnb = GaussianNB()
# Build a Bayesian Classification Model and predict the type using the test
data.
gnb.fit(X_sm_train, y_sm_train)
# Calculate the posteriori probabilities
p = gnb.predict_proba(X_test)
# Predict the target value using the test data.
y_pred_sm = gnb.predict(X_test)

#Time
end = time.time()

# Calculate the accuracy
accuracy = gnb.score(X_test, y_test)
# This is the old formatting method.
# print('Accuracy: {:.4f}'.format(accuracy))
print(f'Accuracy: {accuracy: .4f}')
# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred_sm)
print(metrics.classification_report(y_test,y_pred_sm))
print(f'Total run time for GradientBoosting with Smote Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gnb,x_sm, y_sm,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Naive Bayesian: {mean_score:.4f}')
```

```
Accuracy:  0.2517
              precision    recall  f1-score   support

           0       0.98      0.17      0.29       655
           1       0.12      0.97      0.21        76

    accuracy                           0.25       731
   macro avg       0.55      0.57      0.25       731
```

```
weighted avg       0.89      0.25      0.28       731

Total run time for GradientBoosting with Smote Dataset: 0.07 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Naive Bayesian: 0.5781
```

Feature Selection Dataset

```
#NB with Feature Selection Dataset
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics, datasets

#Time
start = time.time()

#Split Dataset
X_train, X_test, y_train, y_test = train_test_split(X_reduced,y,test_size
=.3,
random_state=1234, stratify=y)
# Create a Bayesian Classifier instance for classification
gnb = GaussianNB()
# Build a Bayesian Classification Model and predict the type using the test
data.
gnb.fit(X_train, y_train)
# Calculate the posteriori probabilities
p = gnb.predict_proba(X_test)
# Predict the target value using the test data.
y_pred = gnb.predict(X_test)

#Time
end = time.time()

# Calculate the accuracy
accuracy = gnb.score(X_test, y_test)
# This is the old formatting method.
# print('Accuracy: {:.4f}'.format(accuracy))
print(f'Accuracy: {accuracy: .4f}')
# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
print(metrics.classification_report(y_test,y_pred))
print(f'Total run time for GradientBoosting with Feature selection Dataset:
{(end - start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gnb,X_reduced, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Naive Bayesian: {mean_score:.4f}')
```

```
Accuracy:  0.2476
              precision    recall  f1-score   support

           0       0.97      0.16      0.28       655
           1       0.12      0.96      0.21        76

    accuracy                           0.25       731
   macro avg       0.55      0.56      0.25       731
weighted avg       0.88      0.25      0.27       731

Total run time for GradientBoosting with Feature selection Dataset: 0.01
seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Naive Bayesian: 0.2623
```

With Original Dataset?

Don't we have this twice?

```
#NB with Original Dataset
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics, datasets

#Time
start = time.time()

#Split Dataset
X_train, X_test, y_train, y_test = train_test_split(Xp,y,test_size =.3,
random_state=1234, stratify=y)
# Create a Bayesian Classifier instance for classification
gnb = GaussianNB()
# Build a Bayesian Classification Model and predict the type using the test
data.
gnb.fit(X_train, y_train)
# Calculate the posteriori probabilities
p = gnb.predict_proba(X_test)
# Predict the target value using the test data.
y_pred = gnb.predict(X_test)

#Time
end = time.time()

# Calculate the accuracy
accuracy = gnb.score(X_test, y_test)
# This is the old formatting method.
# print('Accuracy: {:.4f}'.format(accuracy))
print(f'Accuracy: {accuracy: .4f}')
# Build a confusion matrix and calculate evaluation ratios
cm = metrics.confusion_matrix(y_test,y_pred);cm
```

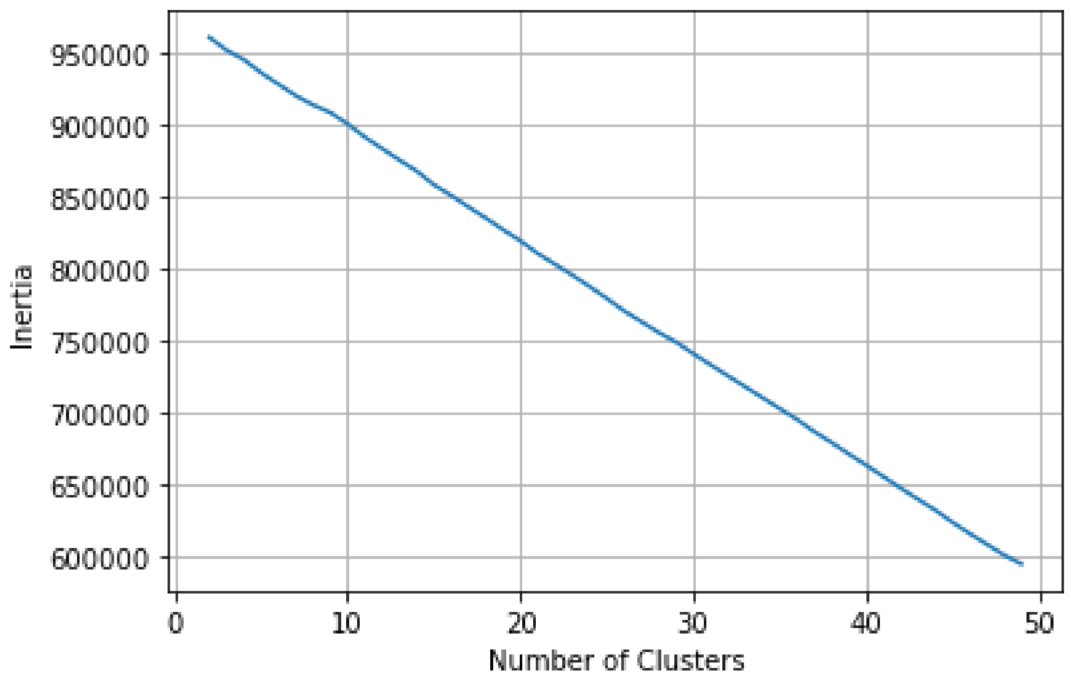
```
print(metrics.classification_report(y_test,y_pred))
print(f'Total run time for GradientBoosting with PCA Dataset: {(end -
start):.2f} seconds')

#k-fold
mean_score = np.mean(cross_val_score(gnb,Xp, y,cv=5))
print('**Mean Score (Accuracy) after applying k-fold**')
print(f'Mean Score for Naive Bayesian: {mean_score:.4f}')
```

```
Accuracy:  0.2531
              precision    recall  f1-score   support

           0       0.98      0.17      0.29       655
           1       0.12      0.97      0.21        76

    accuracy                           0.25       731
   macro avg       0.55      0.57      0.25       731
weighted avg       0.89      0.25      0.28       731

Total run time for GradientBoosting with PCA Dataset: 0.02 seconds
**Mean Score (Accuracy) after applying k-fold**
Mean Score for Naive Bayesian: 0.2434
```

K-Means Clustering - Finding K

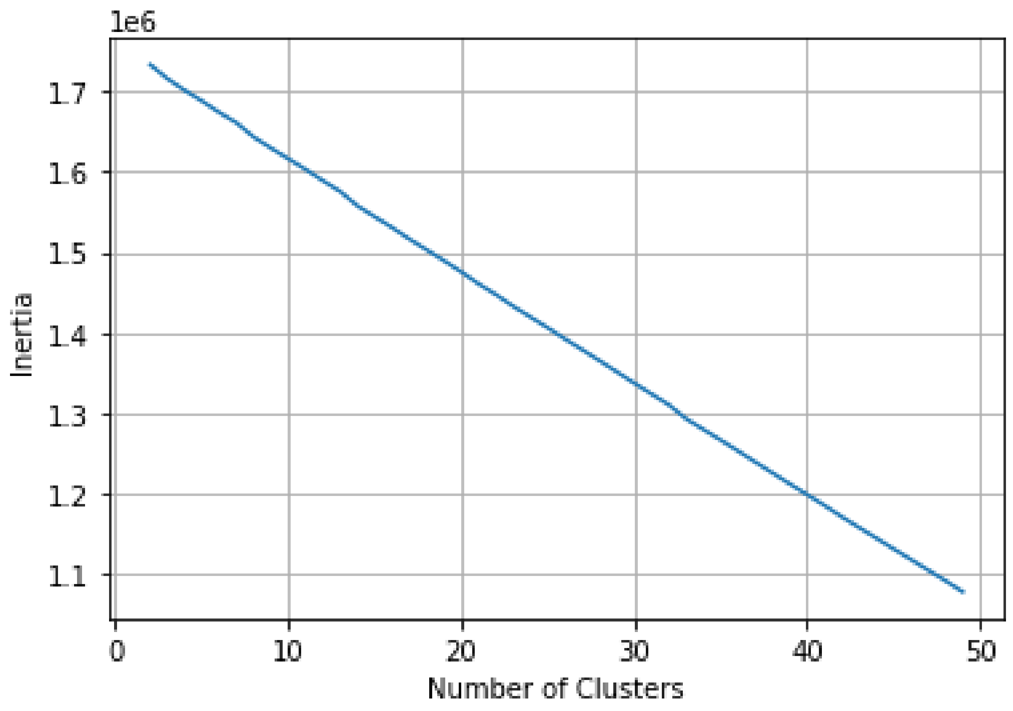
Original Dataset

```
# Using original data
# Z score normalization
scaler = StandardScaler()
Xn = scaler.fit_transform(x)

#Computing value of K using elbow method
# If K value is large, inertia will decrease. Lower the inertia better model
the is.
inertia_list = []
for i in range(2,50):
    km = KMeans(n_clusters=i, random_state=1234)
    km.fit(Xn)
    inertia_list.append(km.inertia_)

# for i in range(len(inertia_list)):
#     print('{0}: {1:.2f}'.format(i+2, inertia_list[i]))

# Scree plot
plt.plot(range(2,50), inertia_list)
plt.grid(True)
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```



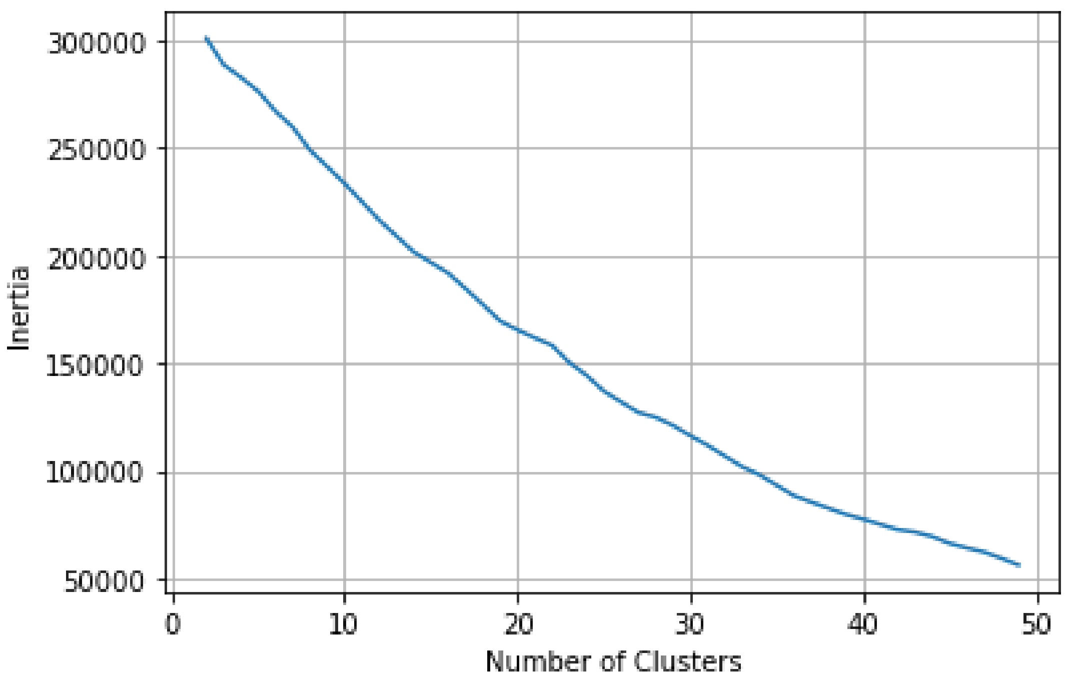
SMOTE Applied Dataset

```
# Using SMOTE applied dataset
# Z score normalization
scaler = StandardScaler()
Xn = scaler.fit_transform(x_sm)

#Computing value of K using elbow method
# If K value is large, inertia will decrease. Lower the inertia better model
the is.
inertia_list = []
for i in range(2,50):
    km = KMeans(n_clusters=i, random_state=1234)
    km.fit(Xn)
    inertia_list.append(km.inertia_)

# for i in range(len(inertia_list)):
#     print('{0}: {1:.2f}'.format(i+2, inertia_list[i]))

# Scree plot
plt.plot(range(2,50), inertia_list)
plt.grid(True)
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```



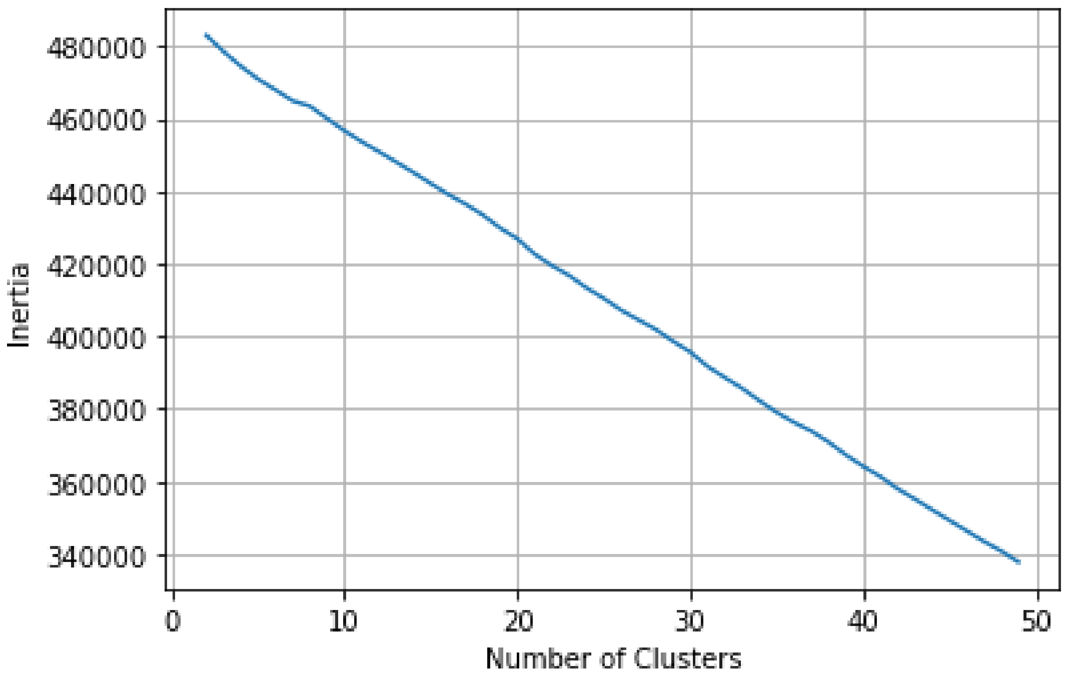
Feature Selection Dataset

```
# Using Feature Selection Dataset
# Z score normalization
scaler = StandardScaler()
Xn = scaler.fit_transform(X_reduced)

#Computing value of K using elbow method
# If K value is large, inertia will decrease. Lower the inertia better model
the is.
inertia_list = []
for i in range(2,50):
    km = KMeans(n_clusters=i, random_state=1234)
    km.fit(Xn)
    inertia_list.append(km.inertia_)

# for i in range(len(inertia_list)):
#     print('{0}: {1:.2f}'.format(i+2, inertia_list[i]))

# Scree plot
plt.plot(range(2,50), inertia_list)
plt.grid(True)
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```



PCA Applied Dataset

```
# Using PCA Applied Dataset
# Z score normalization
scaler = StandardScaler()
Xn = scaler.fit_transform(Xp)

#Computing value of K using elbow method
# If K value is large, inertia will decrease. Lower the inertia better model
the is.
inertia_list = []
for i in range(2,50):
    km = KMeans(n_clusters=i, random_state=1234)
    km.fit(Xn)
    inertia_list.append(km.inertia_)

# for i in range(len(inertia_list)):
#     print('{0}: {1:.2f}'.format(i+2, inertia_list[i]))

# Scree plot
plt.plot(range(2,50), inertia_list)
plt.grid(True)
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.show()
```


480000
460000
440000
420000
400000
380000
360000
340000
Inertia
0
10
20
30
40
50
Number of Clusters