Prediction of Human Health Risks to Agrochemicals using Machine Learning

Ann Maria John, Padmini Arra, Priyanka Bhyregowda, Sahana Thoravalli Prabhuswamy, Supriya Vasagiri

Department of Applied Data Science, San Jose state University

DATA 245: Machine Learning Technologies

Professor Vishnu Pendyala

October 05, 2023

Abstract

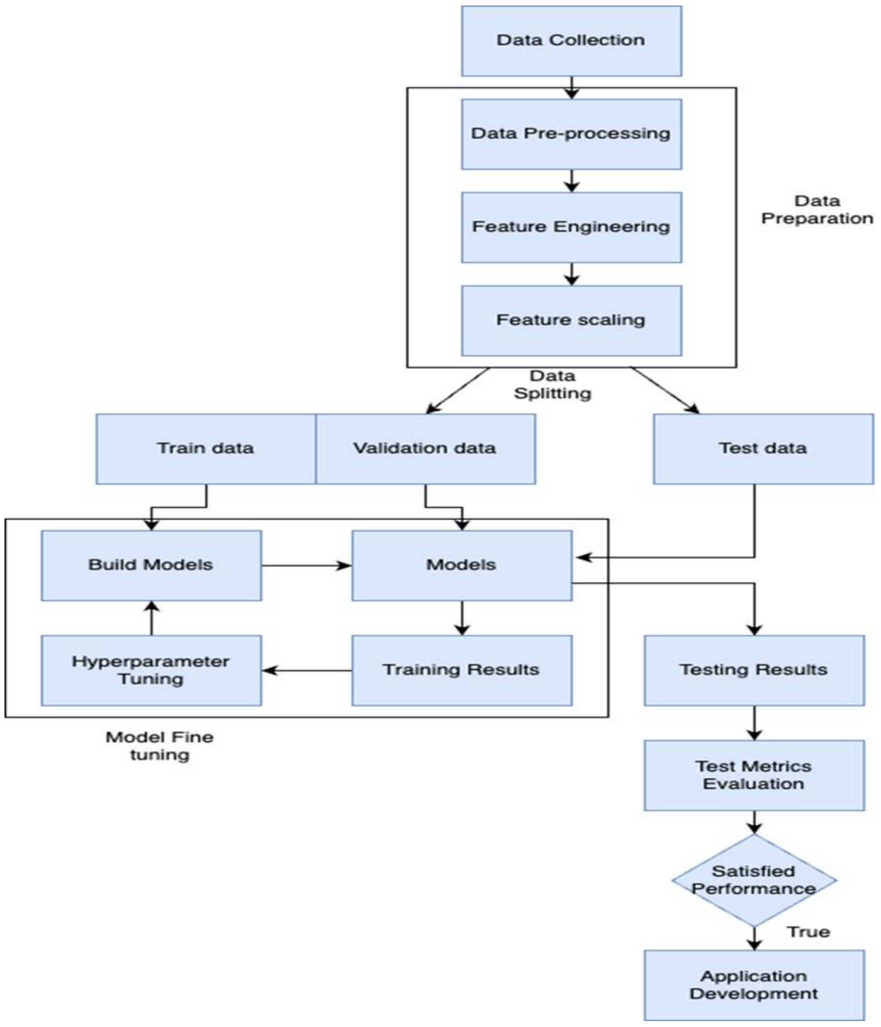
To meet the growing demand for food modern farmer’s use pesticides for crop protection and to increase yield production. However, the widespread use of pesticides in agriculture has many adverse effects on human health as well as environmental sustainability. Exposure to pesticides has many side effects and causes many health problems such as cancers, renal failures, immune suppression, genetic disorders, etc. Harmful chemicals enter the body in many ways, but exposure to these harsh chemicals is more common in people who are involved in agricultural practices. In our study, we have used data from the National Health and Nutrition Examination Survey (NHANES) from 2015 to 2016 across the U.S. from different groups. The main objective of our study is to discern the most effective and optimal model to classify people into higher risk and lower risk of diagnosing renal failure and other health issues using classification models like Linear classifier, Random Forest classifier, KNN, Decision tree classifier, support vector machines (SVM), logistic regression and ensemble techniques. Further, using feature selection like select K-best and getting p-significant features are used in dimension reduction techniques and the results are compared using evaluation metrics like accuracy, f1-score, Recall and precision from all the developed models.

Keywords: Linear classifier, SVM, Logistic regression, Random Forest classifier, PCA

Introduction

Pesticides are widely used in producing food to control pests such as insects, rodents, weeds, bacteria, mold, and fungus. Among these, malathion is a man-made organophosphate insecticide that is commonly used to control mosquitoes and other insects that attack fruits, vegetables, plants, and shrubs. It can also be found in other pesticide products used indoors and in pets to control ticks and insects, such as fleas and ants. These are associated with renal diseases and other health-related problems. This is an invisible threat to most people. With the use of machine learning algorithms, we can uncover hidden patterns and relationships within the data that may not be immediately apparent through traditional statistical analysis. This can lead to more comprehensive and data-driven insights into the risks associated with pesticide exposure. These models can help in detecting early signs of health issues or trends associated with exposure. Early detection can facilitate timely intervention and preventive measures to mitigate health risks. To date, limited studies have focused on the issues of agrochemical use and protective measures that may help to reduce occupational exposure during pesticide application. The current research aims at developing machine learning models to detect pesticide exposure along with exploring different techniques for imputation like constant, KNN imputation, feature selection processes such as p-value significance, SelectKbest, and Recursive Feature Elimination(RFE), Oversampling techniques like SMOTE and ADASYN and various performance metrics such as accuracy, precision, recall and F1 score for all the models. Each of the models is subjected to hyperparameter tuning to get the best metrics.

Figure 1: Experimental Setup



Data Collection:

We did an extensive dataset search on the Google Dataset Search website and IEEE DataPort. We began by conducting a thorough search for datasets pertaining to the UN Sustainable Development Goals. The dataset encompasses information about exposure to pesticides and was downloaded from the National Health and Nutrition Examination Survey (NHANES). There is a total of 39 .csv files related to questionnaires, pesticides, and renal function. We have extracted the required data from each .csv file related to the experiment. The data was pooled from the 2015-

2016 study wave and has details related to pesticides, participant information, and their corresponding lab results. Data was collected with informed consent from the participants using structured questionnaires framed by NHANES. The combination of this descriptive-analytical cross-sectional study with a quantitative approach and laboratory results conducted across the US is used in our study.

Dataset generation

Different self-answered questionnaires of each patient are considered from the NHANES website for 2015-2016 and only the required column related to the experiment is manually extracted from the questionnaire csv's and all the other columns are dropped.

The laboratory files related to agrochemicals and renal function are downloaded from the NHANES website for 2015-2016 and the pesticide concentrations from urine tests from the respective csv's are extracted and merged with the questionnaires data on SEQN which is patient id.Target variable is derived from UACR which is Urine Albumin and creatinine ratio. If the ratio of UACR is >30(mg/g) then it is considered as reduced kidney function.

Preprocessing:

The merged dataset of questionnaires and laboratory data with renal function as our target variable is used in preprocessing. Features like SEQN, and columns with only zeros and 1's are dropped (Drug Addict, Dialysis, asthma, asthma_current, current_chronic bronchitis). All categorical features are converted using dummy encoding and the first column is dropped to avoid overfitting. Feature 'Gender' is mapped to binary values.

Train test Split: We have used Sklearn train_test_split for splitting the data. The data is split into 80:20 ratio.

Standardization:

After the train test split is performed, the train data is fitted and transformed using the Column transformer where the numerical features are standardized using the StandardScaler() function from Sklearn, the binary features are bypassed, and the test data is transformed.

EDA

The dataset has 6221 rows and 79 columns. 78 feature variables and 'UACR' is the target variable. The dataset imbalance is shown in Figure 2. There are 40 columns with missing values which are imputed using different techniques like constant imputation, KNN imputation, and median Imputation. There are 49 features with binary categories, 2 features with more than 2 categories, and 26 features with numerical variables. The 2 multi-category features are encoded using get_dummy encoding and converted into binary categories. The numerical features are standardized to reduce the effect of features with high numerical values. A heatmap was visualized for correlation values of p-significant features as shown in figure 4 and 6. Some exploratory analysis was done on other categorical features as well as shown in Figure 2 and the most significant features in Figure 5.

We did an extensive dataset search on the Google Dataset Search website and IEEE DataPort. We began by conducting a thorough search for datasets pertaining to the UN Sustainable Development Goals. The dataset encompasses information about exposure to pesticides and was downloaded from the National Health and Nutrition Examination Survey (NHANES). There is a total of 39 .csv files related to questionnaires, pesticides, and renal function. We have extracted the required data from each .csv file related to the experiment. The data was pooled from the 2015-2016 study wave and has details related to pesticides, participant information, and their corresponding lab results. Data was collected with informed consent from the participants using structured questionnaires framed by NHANES. The combination of this descriptive-analytical cross-sectional study with a quantitative approach and laboratory results conducted across the US is used in our study.

Dataset generation

Different self-answered questionnaires of each patient are considered from the NHANES website for 2015-2016 and only the required column related to the experiment is manually extracted from the questionnaire csv's and all the other columns are dropped.

The laboratory files related to agrochemicals and renal function are downloaded from the NHANES website for 2015-2016 and the pesticide concentrations from urine tests from the respective csv's are extracted and merged with the questionnaires data on SEQN which is patient id.

Target variable is derived from UACR which is Urine Albumin and creatinine ratio. If the ratio of UACR is >30(mg/g) then it is considered as reduced kidney function.

Preprocessing:

The merged dataset of questionnaires and laboratory data with renal function as our target variable is used in preprocessing. Features like SEQN, and columns with only zeros and 1's are dropped (Drug Addict, Dialysis, asthma, asthma_current, current_chronic bronchitis). All categorical features are converted using dummy encoding and the first column is dropped to avoid overfitting. Feature ‘Gender’ is mapped to binary values.

Train test Split:

Test train split: We have used Sklearn train_test_split for splitting the data. The data is split into 80:20 ratio.

Standardization:

After the train test split is performed, the train data is fitted and transformed using the Column transformer where the numerical features are standardized using the StandardScaler() function from Sklearn, the binary features are bypassed, and the test data is transformed.

EDA

The dataset has 6221 rows and 79 columns. 78 feature variables and ‘UACR’ is the target variable. The dataset imbalance is shown in Figure 2. There are 40 columns with missing values which are imputed using different techniques like constant imputation, KNN imputation, and median Imputation. There are 49 features with binary categories, 2 features with more than 2 categories, and 26 features with numerical variables. The 2 multi-category features are encoded using get_dummy encoding and converted into binary categories. The numerical features are standardized to reduce the effect of features with high numerical values. A heatmap was visualized for correlation values of p-significant features as shown in figure 4 and 6. Some exploratory analysis was done on other categorical features as well as shown in Figure 2 and the most significant features in Figure 5.

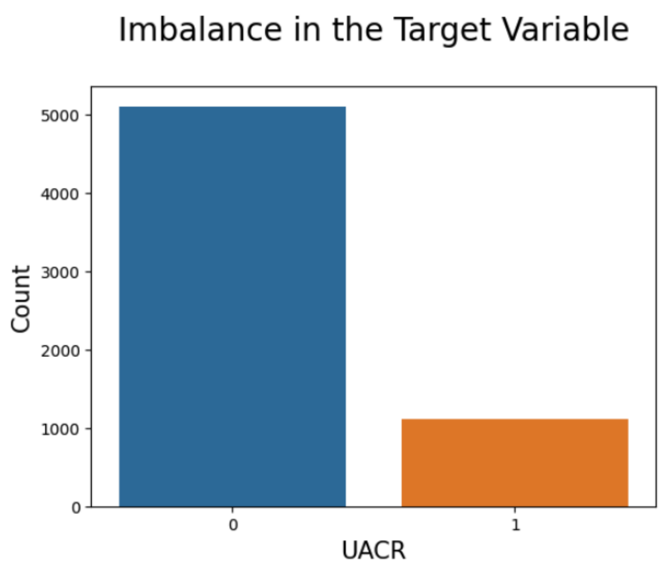


Figure 2:Imbalance in the target variable

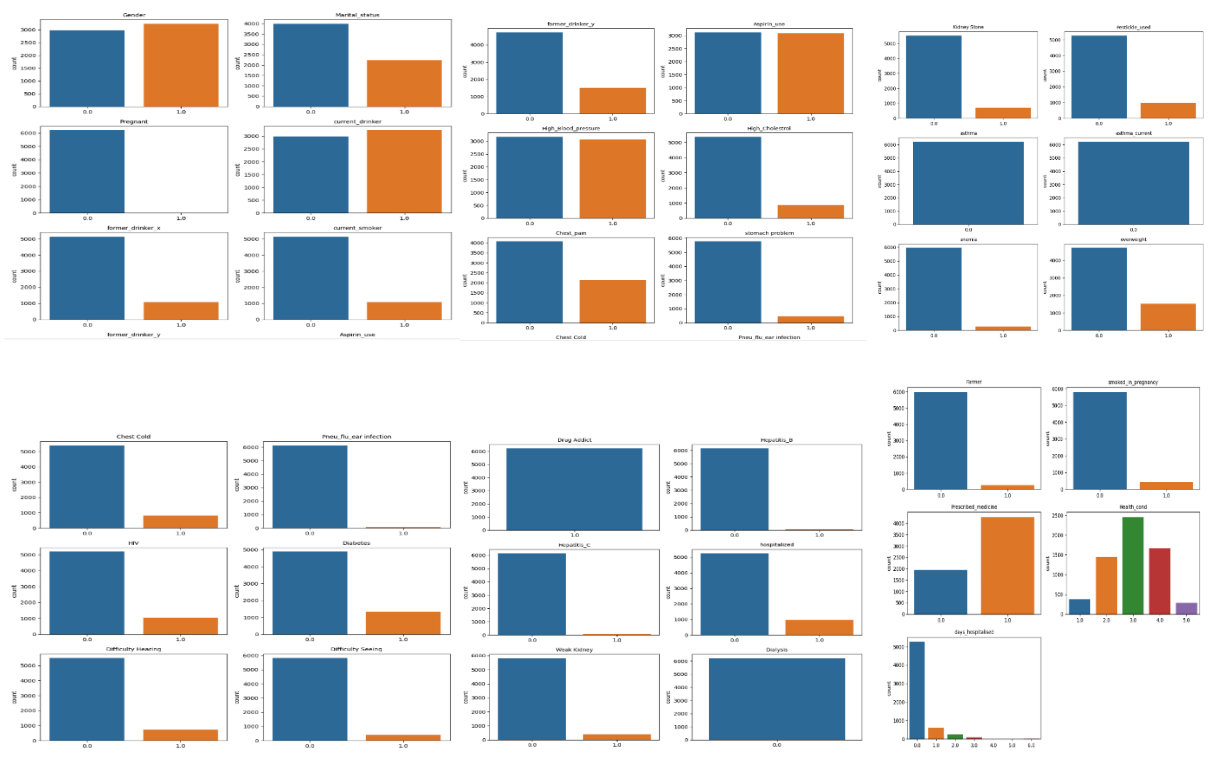
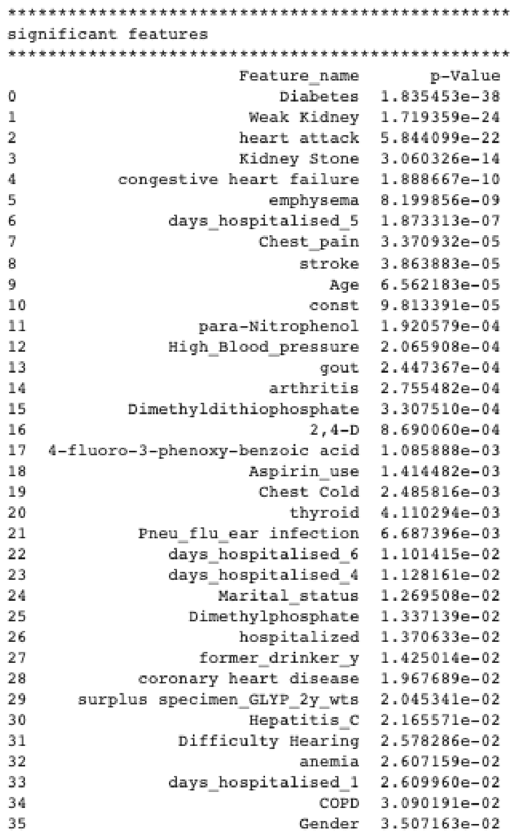


Figure 3:EDA on a few categorical variables



```
****************************************************
significant features
****************************************************
                          Feature_name       p-Value
0                             Diabetes  1.835453e-38
1                          Weak Kidney  1.719359e-24
2                         heart attack  5.844099e-22
3                         Kidney Stone  3.060326e-14
4             congestive heart failure  1.888667e-10
5                            emphysema  8.199856e-09
6                    days_hospitalised_5  1.873313e-07
7                           Chest_pain  3.370932e-05
8                               stroke  3.863883e-05
9                                  Age  6.562183e-05
10                               const  9.813391e-05
11                   para-Nitrophenol  1.920579e-04
12                 High_Blood_pressure  2.065908e-04
13                                gout  2.447367e-04
14                           arthritis  2.755482e-04
15             Dimethyldithiophosphate  3.307510e-04
16                               2,4-D  8.690060e-04
17    4-fluoro-3-phenoxy-benzoic acid  1.085888e-03
18                          Aspirin_use  1.414482e-03
19                          Chest Cold  2.485816e-03
20                             thyroid  4.110294e-03
21                Pneu_flu_ear infection  6.687396e-03
22                 days_hospitalised_6  1.101415e-02
23                 days_hospitalised_4  1.128161e-02
24                      Marital_status  1.269508e-02
25                  Dimethylphosphate  1.337139e-02
26                        hospitalized  1.370633e-02
27                    former_drinker_y  1.425014e-02
28               coronary heart disease  1.967689e-02
29        surplus specimen_GLYP_2y_wts  2.045341e-02
30                         Hepatitis_C  2.165571e-02
31                  Difficulty Hearing  2.578286e-02
32                              anemia  2.607159e-02
33                 days_hospitalised_1  2.609960e-02
34                                COPD  3.090191e-02
35                              Gender  3.507163e-02
```

Figure 4:. Features with p-value < 0.05

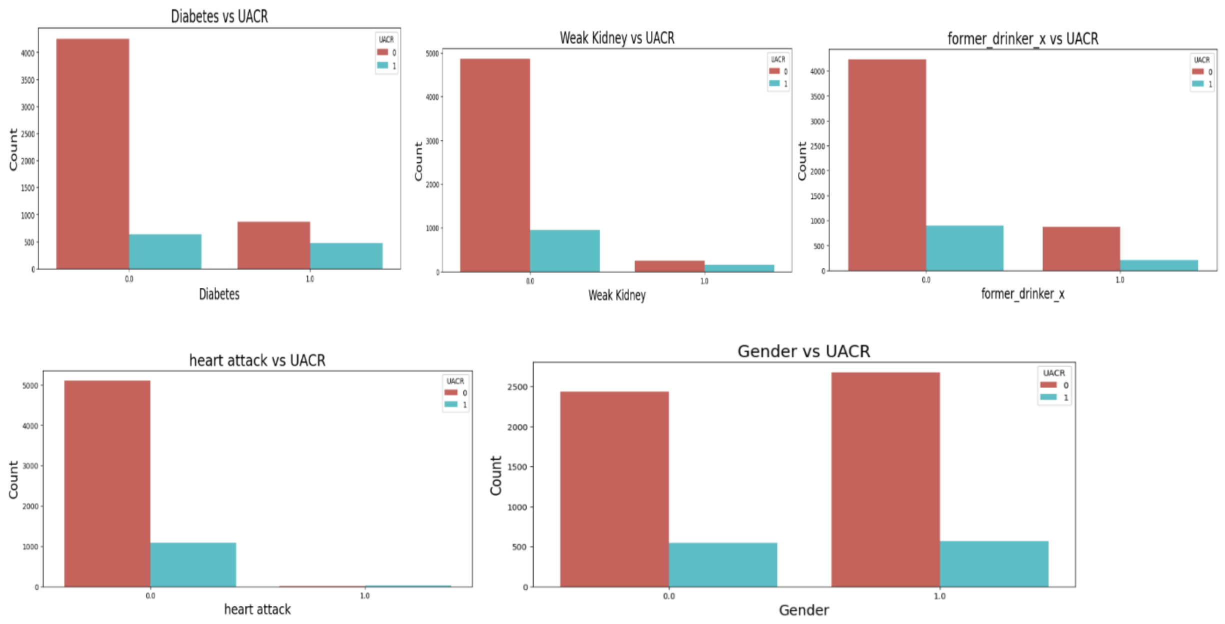


Figure 5:Side-by-side bar plot for significant features VS target variable UACR

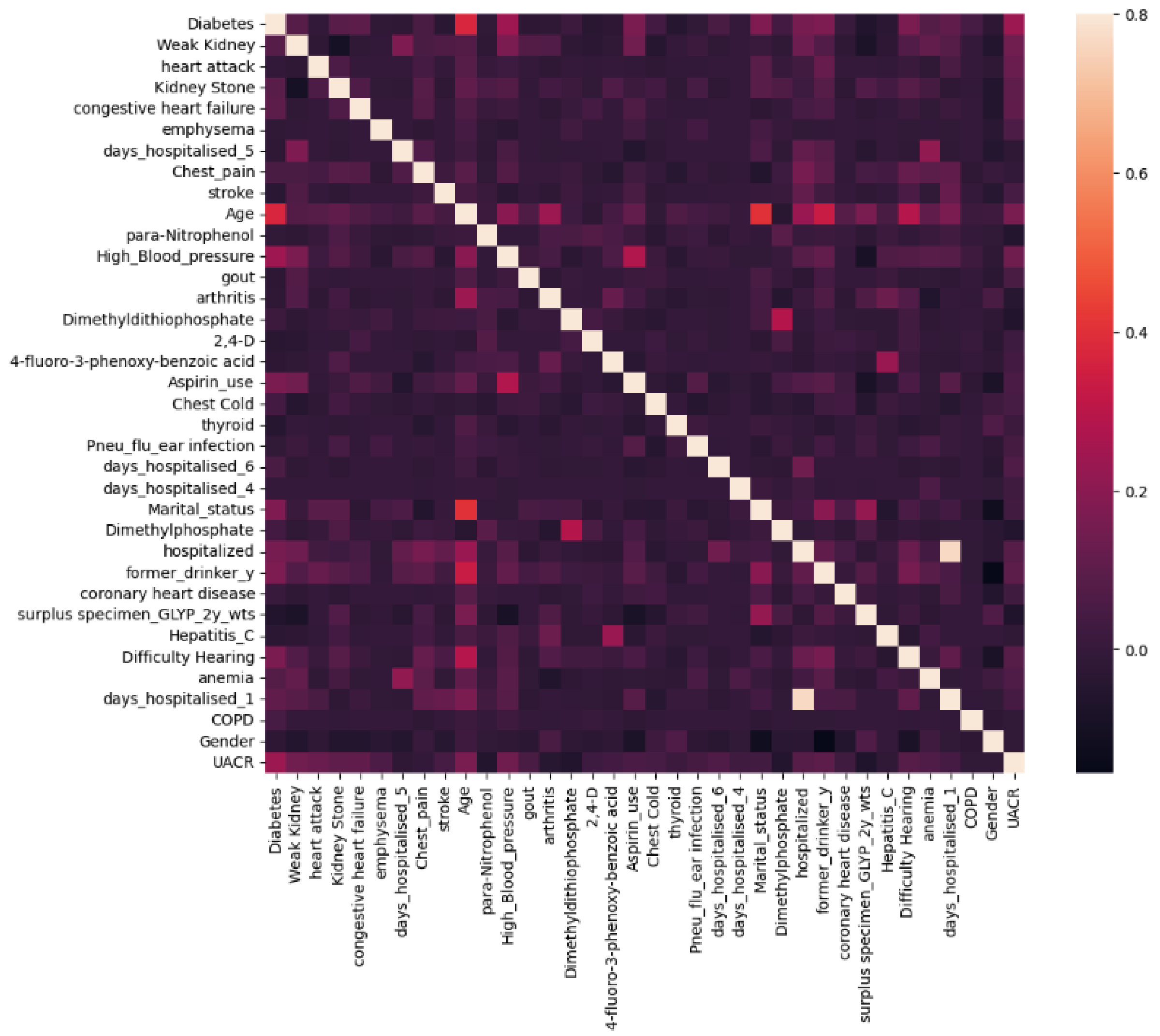


Figure 6:Heat map of correlation values of p-significant features with target variable

Feature Selection

How we select features plays a significant role in improving machine learning algorithms. Three different approaches were used for feature selection.

- **P-value significant feature selection** - Ordinary least square method from stats model is used to determine the p-values for each feature and features with p-values above 0.05 are ignored because of lack of significance. Figure 3 shows the significant features.

- **SelectKbest** - mutual_info_classif parameter is used to find the dependency between features and target variables and rank them to select the k best out of it.
- **Recursive feature elimination** - RandomForest estimator is used to recursively fit the model with a smaller set of features each time and get the feature importance. The process is repeated until we get the desired number of features.

Dimensionality Reduction

A Few dimensionality reduction techniques like FAMD (Factor Analysis for Mixed Data) and Kernel PCA (Principal Component Analysis) were employed in the process of model development. Since the results obtained after applying the dimension reduction techniques were poor, these techniques were not used during the hyperparameter tuning of various models.

Feature encoding

In our study, we have used three encoding techniques like, binary features, categorical features, and numerical features. The binary features are mapped to the numerical values of 0 and 1 while preserving the actual data. Apart from that, categorical features were also used, and a dummy value was used for encoding for the features that have more than two categories. We have also created binary columns for each category within a feature. So, by this, each of the binary columns can be used to represent whether the specific category is present or not. Besides this, for numerical features, we used standardization techniques. It is important to use standardization techniques when we are using Machine learning algorithms such as K nearest neighbors(KNN) and Support Vector Machines(SVM) as these algorithms are sensitive to the scale of input features. So, for that, we have used the StandardScaler() function which is used for transforming the numerical features in such a way that the features that consist of large numerical values will not be impacted while training the model.

Missing data imputation

Missing data is common in all study and survey designs. Partially filled-up surveys could be a possible reason for survey or questionnaire-based data collection. Incomplete medical records are yet another reason for missing data. We use three methods for missing value imputation namely, imputing with constant, imputing with median, and KNN imputation algorithm.

Replacing with constant value

A constant value of -1 was used to replace the missing values in the dataset. Using constant helps simplify the analysis and helps to maintain the consistency of the dataset. This also prevents the introduction of bias into the analysis.

Replacing with median value

Median is the middlemost value. 'fillna' method was used to impute missing data using median.

k-Nearest Neighbor (KNN) Imputer

KNN imputer substitutes missing values using distance functions. KNN works well with non-linear data. Moreover, when compared to other imputers or imputation techniques KNN may provide better performance. It works well on small to medium-sized datasets owing to the computational cost associated with it.

Oversampling

In our dataset there is an Imbalance which refers to a situation where the distribution of classes is not equal. The dataset had approximately 5,000 instances where the target variable urine albumin-creatinine ratio(UACR) indicates no effect and about 1,000 instances where it indicates an effect due to pesticide exposure.

When there is an imbalance in the dataset there are two techniques that can be used to address the scenario. They are oversampling and under sampling. Oversampling involves increasing the number of instances of the minority class to balance the class distribution. Under sampling involves reducing the number of instances of the majority class to balance the class distribution. We choose to oversample as it often suggests improving model performance when the target class has limited data for the minority class.

Synthetic Minority Oversampling Technique (SMOTE)

For each minority class instance, SMOTE generates a synthetic example by considering its k-nearest neighbors(KNN) and creating new instances along the line segments connecting the instance with its neighbors.

smoteNC

Smote extends the capability to handle datasets with a mix of numerical and continuous features. For numerical features, it uses a similar approach to the original SMOTE by interpolating between existing instances. For categorical features, it handles them in a way that ensures the synthetic instances preserve the categorical characteristics.

Borderline-SMOTE

It identifies instances that are misclassified or near the decision boundary between minority and majority classes and applies SMOTE to them. (Kim et al., 2022) explains after applying the KNN algorithm to the minority class data, the Borderline-SMOTE algorithm classifies instances that have more than half of their nearest neighbors belonging to the majority class as borderline instances.

ADASYN (Adaptive Synthetic Sampling)

ADASYN applies the KNN algorithm in the minority class that are difficult to classify based on the density of their local neighborhood. Instances in regions with fewer instances are considered to be more challenging. More synthetic samples are generated in these challenging regions to address the imbalance.

For each model (Logistic Regression, Support Vector Machine, Decision Tree, K-Nearest Neighbors, Random Forest), we applied various SMOTE (Synthetic Minority Over-sampling Technique) methods after imputing missing values, conducting significance testing, and performing recursive feature elimination, to balance the target class distribution for each model individually. This allowed us to assess the effects of different SMOTE variations on the model performance metrics, including accuracy, precision, recall, and F1-score.

Methods

Logistic Regression

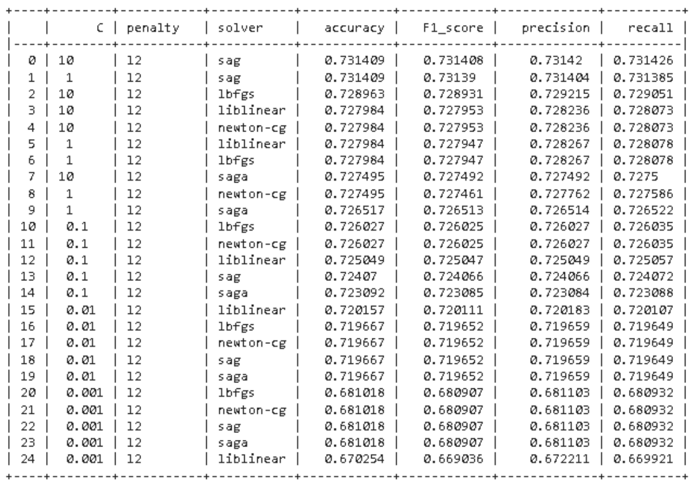
The Logistic Regression algorithm is used for binary classification, predicting the probability that an instance belongs to a particular class. It's widely employed in machine learning for its simplicity and interpretability. Despite its name, Logistic Regression is used for classification rather than regression tasks. We considered this model as a popular choice for introductory classification tasks and as a baseline model in more complex scenarios.

Hyperparameter Tuning

In the process of hyperparameter tuning, various scenarios were explored on the dataset. Table 1. shows promising results achieved by tuning hyperparameters directly after missing value imputation on the data and including the use of Principal Components (p) as well as incorporating the SMOTE-NC technique for data augmentation, was investigated.

The optimal hyperparameter configuration was identified as follows: regularization parameter (C) set to 10, penalty term specified as L2 (due to compatibility constraints with the selected solvers—SAG and LBFGS—where L1 was not applicable), and the solver chosen as sag. This configuration yielded the best performance among the various hyperparameter tuning scenarios explored in this study.

Table 1:Accuracy, F1 score, Precision, and Recall for different hyperparameters.



	C	penalty	solver	accuracy	F1_score	precision	recall
0	10	l2	sag	0.731409	0.731408	0.73142	0.731426
1	1	l2	sag	0.731409	0.73139	0.731404	0.731385
2	10	l2	lbfgs	0.728963	0.728931	0.729215	0.729051
3	10	l2	liblinear	0.727984	0.727953	0.728236	0.728073
4	10	l2	newton-cg	0.727984	0.727953	0.728236	0.728073
5	1	l2	liblinear	0.727984	0.727947	0.728267	0.728078
6	1	l2	lbfgs	0.727984	0.727947	0.728267	0.728078
7	10	l2	saga	0.727495	0.727492	0.727492	0.7275
8	1	l2	newton-cg	0.727495	0.727461	0.727762	0.727586
9	1	l2	saga	0.726517	0.726513	0.726514	0.726522
10	0.1	l2	lbfgs	0.726027	0.726025	0.726027	0.726035
11	0.1	l2	newton-cg	0.726027	0.726025	0.726027	0.726035
12	0.1	l2	liblinear	0.725049	0.725047	0.725049	0.725057
13	0.1	l2	sag	0.72407	0.724066	0.724066	0.724072
14	0.1	l2	saga	0.723092	0.723085	0.723084	0.723088
15	0.01	l2	liblinear	0.720157	0.720111	0.720183	0.720107
16	0.01	l2	lbfgs	0.719667	0.719652	0.719659	0.719649
17	0.01	l2	newton-cg	0.719667	0.719652	0.719659	0.719649
18	0.01	l2	sag	0.719667	0.719652	0.719659	0.719649
19	0.01	l2	saga	0.719667	0.719652	0.719659	0.719649
20	0.001	l2	lbfgs	0.681018	0.680907	0.681103	0.680932
21	0.001	l2	newton-cg	0.681018	0.680907	0.681103	0.680932
22	0.001	l2	sag	0.681018	0.680907	0.681103	0.680932
23	0.001	l2	saga	0.681018	0.680907	0.681103	0.680932
24	0.001	l2	liblinear	0.670254	0.669036	0.672211	0.669921

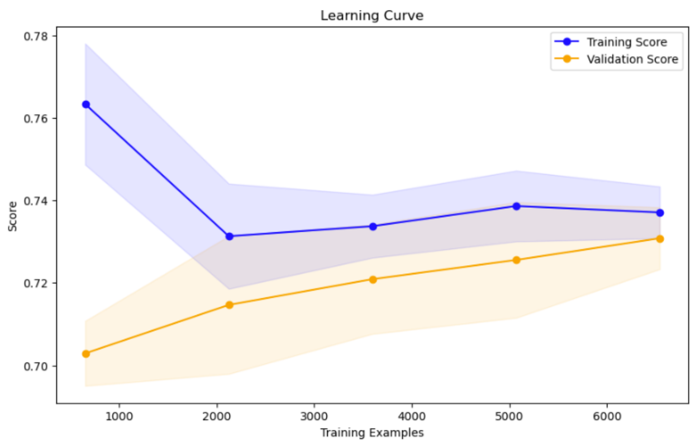


Figure 7: Learning Curve Logistic Regression

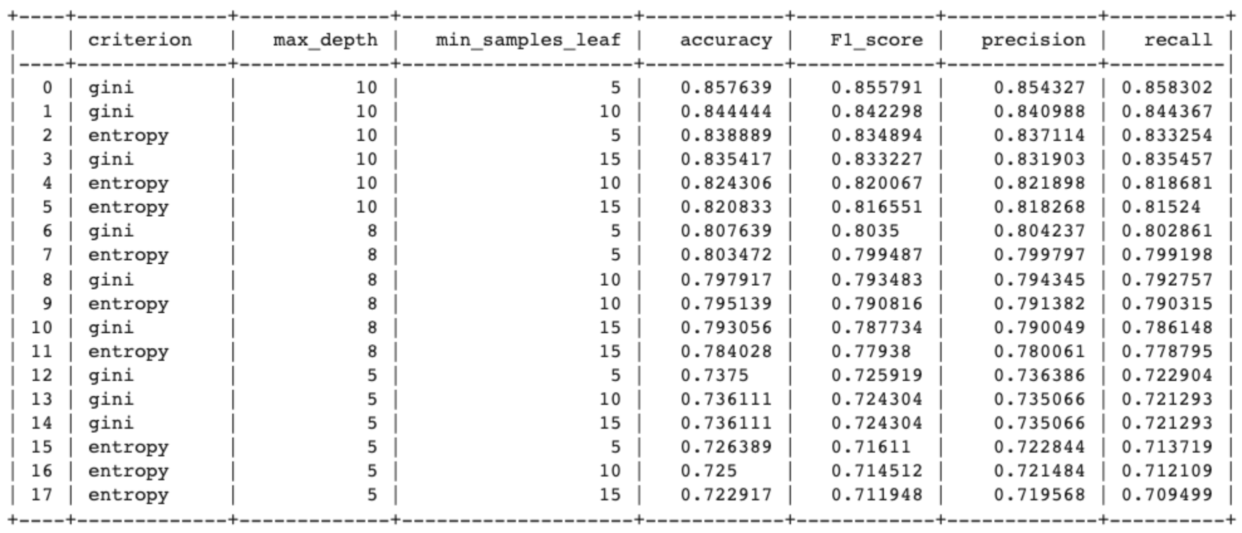
A learning curve with accuracy score is obtained for different numbers of training samples and validation samples as shown in Figure 7. Table 2 compares various cross-validated train data and test data metrics.

Decision Tree

Decision trees are supervised non-parametric machine learning algorithms. It uses impurity criteria such as Gini and entropy to determine the best split to divide the classes. A base model of a decision tree with default hyperparameters is first generated on the dataset obtained after selecting p-value significant features and RFE feature selection with 30 features. The Fully grown tree is shown in Figure 6. Hyperparameter tuning is performed with the following parameters of max_depth : [5,8,10], min_sample_leaf : [5,10,15] and, criterion : ['gini', 'entropy'].

Table 1 shows accuracy, f1 score, precision, and recall for different hyperparameters. Trees with a depth greater than 15 and min_sample_leaf less than 5 generated an overfitting model with higher train accuracy and lower test accuracy. The optimal parameters were found to be max_depth = 10, min_sample_leaf = 5, and criterion = gini.

Table 2:Accuracy, F1 score, Precision, and Recall for different hyperparameters.



```
+----+-------------+-------------+--------------------+------------+------------+-------------+----------+
|    | criterion   |   max_depth |   min_samples_leaf |   accuracy |   F1_score |   precision |   recall |
|----+-------------+-------------+--------------------+------------+------------+-------------+----------|
|  0 | gini        |          10 |                  5 |   0.857639 |   0.855791 |    0.854327 | 0.858302 |
|  1 | gini        |          10 |                 10 |   0.844444 |   0.842298 |    0.840988 | 0.844367 |
|  2 | entropy     |          10 |                  5 |   0.838889 |   0.834894 |    0.837114 | 0.833254 |
|  3 | gini        |          10 |                 15 |   0.835417 |   0.833227 |    0.831903 | 0.835457 |
|  4 | entropy     |          10 |                 10 |   0.824306 |   0.820067 |    0.821898 | 0.818681 |
|  5 | entropy     |          10 |                 15 |   0.820833 |   0.816551 |    0.818268 | 0.81524  |
|  6 | gini        |           8 |                  5 |   0.807639 |   0.8035   |    0.804237 | 0.802861 |
|  7 | entropy     |           8 |                  5 |   0.803472 |   0.799487 |    0.799797 | 0.799198 |
|  8 | gini        |           8 |                 10 |   0.797917 |   0.793483 |    0.794345 | 0.792757 |
|  9 | entropy     |           8 |                 10 |   0.795139 |   0.790816 |    0.791382 | 0.790315 |
| 10 | gini        |           8 |                 15 |   0.793056 |   0.787734 |    0.790049 | 0.786148 |
| 11 | entropy     |           8 |                 15 |   0.784028 |   0.77938  |    0.780061 | 0.778795 |
| 12 | gini        |           5 |                  5 |   0.7375   |   0.725919 |    0.736386 | 0.722904 |
| 13 | gini        |           5 |                 10 |   0.736111 |   0.724304 |    0.735066 | 0.721293 |
| 14 | gini        |           5 |                 15 |   0.736111 |   0.724304 |    0.735066 | 0.721293 |
| 15 | entropy     |           5 |                  5 |   0.726389 |   0.71611  |    0.722844 | 0.713719 |
| 16 | entropy     |           5 |                 10 |   0.725    |   0.714512 |    0.721484 | 0.712109 |
| 17 | entropy     |           5 |                 15 |   0.722917 |   0.711948 |    0.719568 | 0.709499 |
+----+-------------+-------------+--------------------+------------+------------+-------------+----------+
```

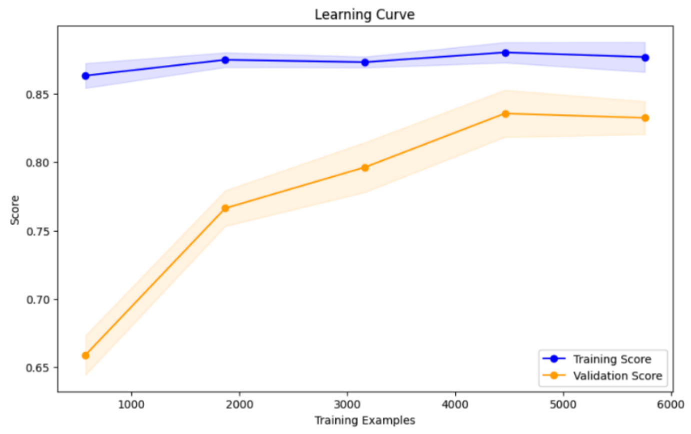


Figure 8. Decision Tree model learning curve using train and validation accuracy scores.

A learning curve with accuracy score is obtained for different numbers of training samples and validation samples as shown in Figure 8.

Support Vector Classifier

A Support Vector Classifier (SVC) is a type of supervised machine learning algorithm that falls under the broader category of Support Vector Machines (SVMs). SVMs are used for classification and regression analysis. The primary objective of an SVM is to find a hyperplane in a high-dimensional space that best separates data points of different classes.

A base model of support vector classifier with default hyperparameters was generated after selecting k best features with k equal to 40. ADASYN oversampling technique was applied to the selected features. Hyperparameter tuning was performed on regularization hyperparameter, C and kernel. C value of ‘100’ and ‘rbf’ kernel were obtained as the optimal hyper parameters. Hyper parameters considered included C : [1, 10, 100] and kernel : [‘linear’, ‘rbf’]. Cross validation was performed using the optimal hyper parameters, and was tested on the testing data. Table 3 below shows the accuracy, F1 score, precision and recall during hyperparameter tuning.

Table 3:SVC hyperparameter tuning.

```
+----+----------------------------+----------+------------+------------+-------------+----------+
|    |   Regularization_parameter | Kernel   |   accuracy |   F1_score |   precision |   recall |
|----+----------------------------+----------+------------+------------+-------------+----------|
|  0 |                        100 | rbf      |   0.933706 |   0.93289  |    0.930439 | 0.940368 |
|  1 |                         10 | rbf      |   0.878576 |   0.876985 |    0.874881 | 0.88319  |
|  2 |                          1 | rbf      |   0.775994 |   0.770176 |    0.769995 | 0.770365 |
+----+----------------------------+----------+------------+------------+-------------+----------+
```

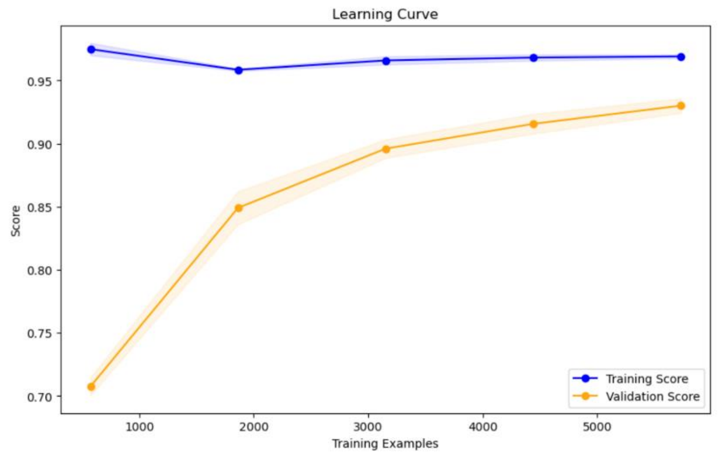


Figure 9:Figure SVC learning curve

SVC learning curve with accuracy score shows an almost steady training score and an increasing validation score.

Random Forest

Random Forest is an ensemble method using many decision trees and bootstrap aggregation. After each decision tree in the random forest predicts renal risk, random forest considers the votes of the prediction. In our study we have considered the best hyperparameter max depth=10 and n_estimators=100. Table shows the f1-score for

The optimal parameters for random forest is tuned and we have selected the max-depth of 10 and entropy as our hyperparameter in the final code as shown below.

```
2 gini 0.4476486246672582
4 gini 0.4607975255999658
6 gini 0.5664690052469303
8 gini 0.7299888401994115
10 gini 0.7907200038729406

gini_index optimal_hyperparameters= (10, 'gini', 0.7907200038729406)
2 entropy 0.4476486246672582
4 entropy 0.49029390026520586
6 entropy 0.5484547990697339
8 entropy 0.6907545069309775
10 entropy 0.7907200038729406

entropy optimal_hyperparameters= (10, 'entropy', 0.7907200038729406)
```

The learning curve for max depth of 10 and n_estimators of 100 is selected and the learning curve is obtained for train data.

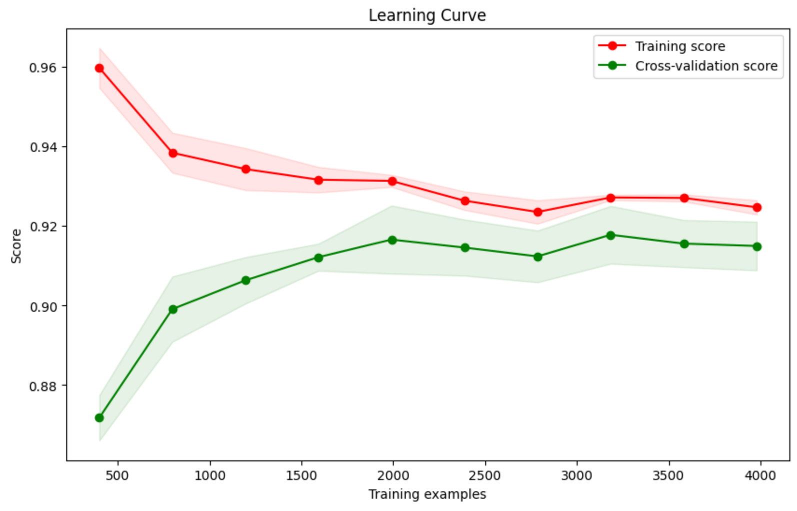
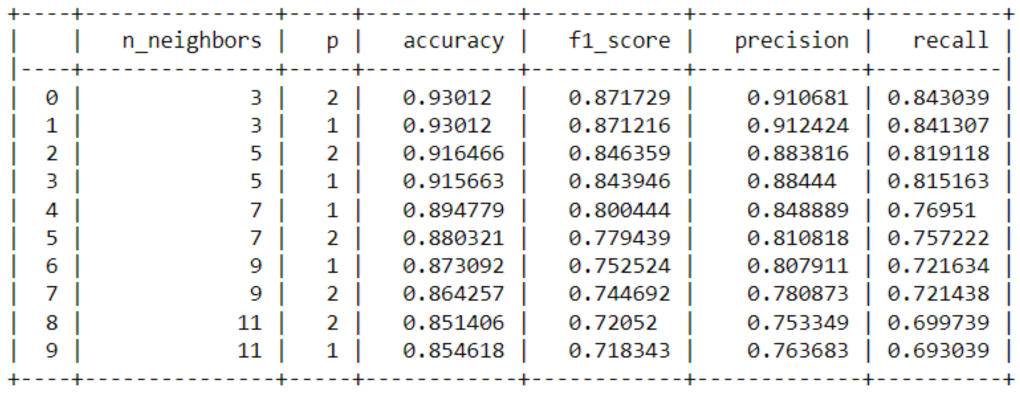


Figure 10:Random Forest learning curve

KNN

K-Nearest Neighbors (KNN) algorithm is one of the simple and effective nonparametric method which is used for classification and regression tasks. Generally, depending upon the majority class of it's neighboring samples, this algorithm is used to predict the class of the given sample. It is used to make predictions on the new sample depending upon the k-nearest neighbors along with it’s class labels in a feature space.

The hyperparameters that were considered included the number of nearest neighbors and distance metrics like manhattan distance and euclidean distance were taken into consideration. These distance metrics were used to find the proximity between the data points in the feature space. So the hyperparameter tuning process involves finding the different k values and also selecting the optimal distance metrics.



```
+----+---------------+-----+------------+------------+-------------+----------+
|    |   n_neighbors |   p |   accuracy |   f1_score |   precision |   recall |
|----+---------------+-----+------------+------------+-------------+----------|
|  0 |             3 |   2 |   0.93012  |   0.871729 |    0.910681 | 0.843039 |
|  1 |             3 |   1 |   0.93012  |   0.871216 |    0.912424 | 0.841307 |
|  2 |             5 |   2 |   0.916466 |   0.846359 |    0.883816 | 0.819118 |
|  3 |             5 |   1 |   0.915663 |   0.843946 |    0.88444  | 0.815163 |
|  4 |             7 |   1 |   0.894779 |   0.800444 |    0.848889 | 0.76951  |
|  5 |             7 |   2 |   0.880321 |   0.779439 |    0.810818 | 0.757222 |
|  6 |             9 |   1 |   0.873092 |   0.752524 |    0.807911 | 0.721634 |
|  7 |             9 |   2 |   0.864257 |   0.744692 |    0.780873 | 0.721438 |
|  8 |            11 |   2 |   0.851406 |   0.72052  |    0.753349 | 0.699739 |
|  9 |            11 |   1 |   0.854618 |   0.718343 |    0.763683 | 0.693039 |
+----+---------------+-----+------------+------------+-------------+----------+
```

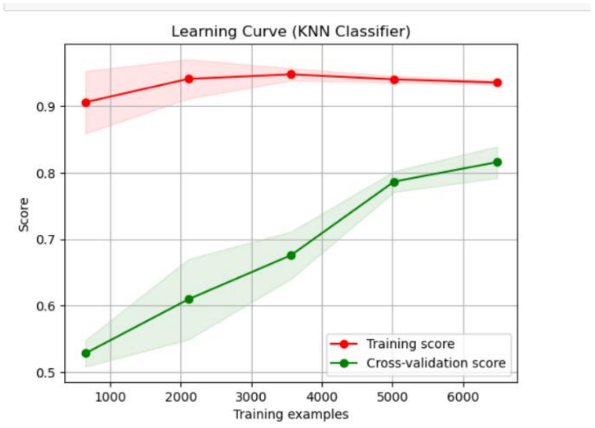


Figure 11:KNN learning curve

Results

Evaluation Metrics

Owing to the classification nature of our problem, the evaluation metrics under consideration is derived from the confusion matrix. Confusion matrix used in our project is as shown in table**1.**

	Predicted		
		Low kidney disfunction	High kidney disfunction
Actual	Low kidney disfunction	True Negative(TN)	False Positives(FP)
	High kidney disfunction	False Negatives(FN)	True positives(TP)

The dataset is a binary classification problem and has a class imbalance, so we consider precision, recall, auc-roc score,f1-Score over Accuracy,.

Recall shows the rate of patients predicted kidney risks among actual patients with renal risks, which can be calculated using equation(1).

Equation 1

$$Recall = \frac{TP}{TP + FN}$$

Precision shows the rate of actual kidney risks among the patients with renal risks, which can be calculated using equation(2).

Equation 2

$$Precision = \frac{TP}{TP + FP}$$

The F1-Score uses both precision and recall, for its calculation which is shown in equation(3)

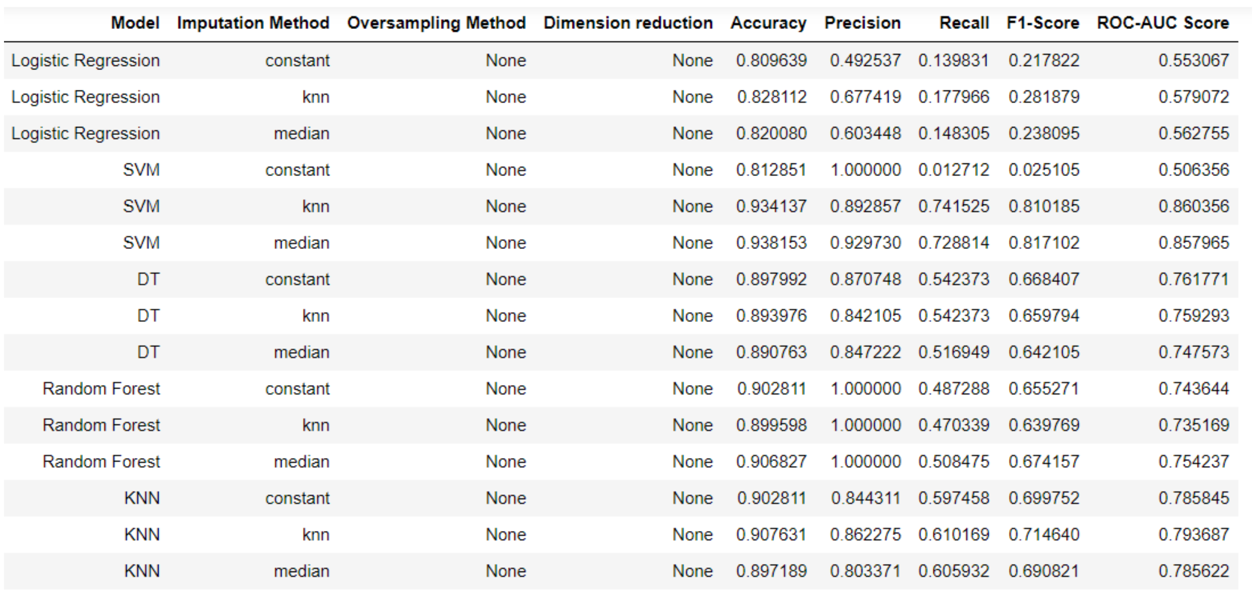
Equation 3

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

Evaluation of Imputation

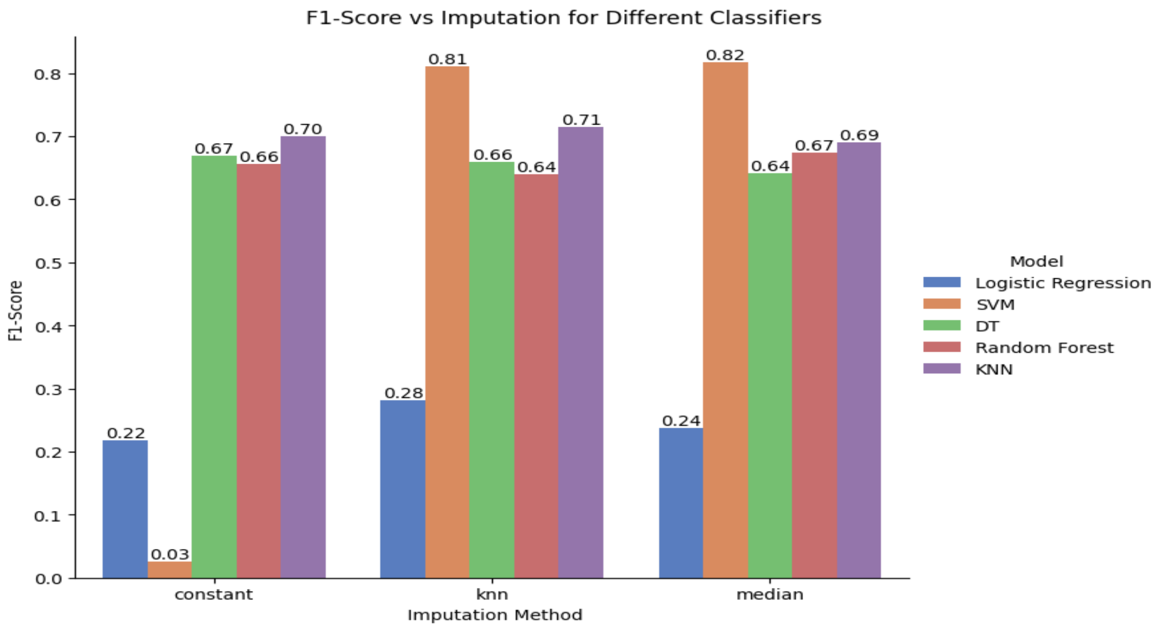
After manual feature selection from the list of questionnaires from NHANES the final data frame is imputed with different imputation methods like KNN , constant and median imputation. The dataset is separated into train test data in the ratio of 80:20 to build and test the model. The performances of different imputation methods are compared for different algorithms of machine learning as shown in the figure**8** .

From the experiment we found that KNN imputation performed slightly better than the other two methods.



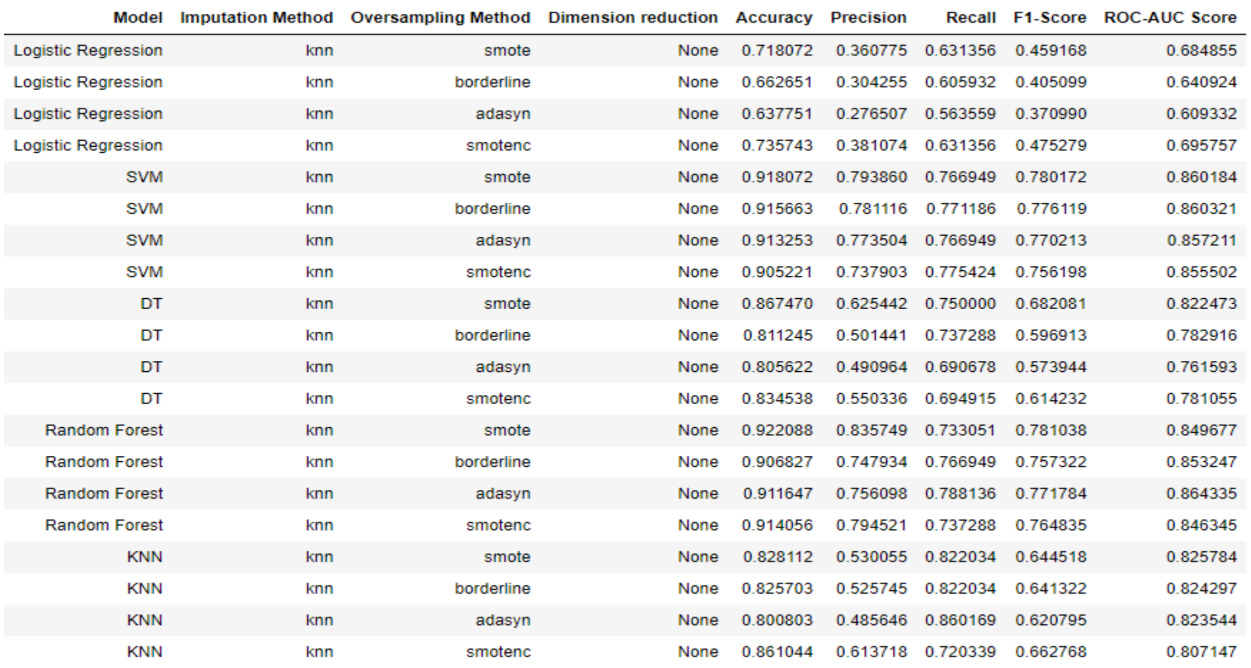
Model	Imputation Method	Oversampling Method	Dimension reduction	Accuracy	Precision	Recall	F1-Score	ROC-AUC Score
Logistic Regression	constant	None	None	0.809639	0.492537	0.139831	0.217822	0.553067
Logistic Regression	knn	None	None	0.828112	0.677419	0.177966	0.281879	0.579072
Logistic Regression	median	None	None	0.820080	0.603448	0.148305	0.238095	0.562755
SVM	constant	None	None	0.812851	1.000000	0.012712	0.025105	0.506356
SVM	knn	None	None	0.934137	0.892857	0.741525	0.810185	0.860356
SVM	median	None	None	0.938153	0.929730	0.728814	0.817102	0.857965
DT	constant	None	None	0.897992	0.870748	0.542373	0.668407	0.761771
DT	knn	None	None	0.893976	0.842105	0.542373	0.659794	0.759293
DT	median	None	None	0.890763	0.847222	0.516949	0.642105	0.747573
Random Forest	constant	None	None	0.902811	1.000000	0.487288	0.655271	0.743644
Random Forest	knn	None	None	0.899598	1.000000	0.470339	0.639769	0.735169
Random Forest	median	None	None	0.906827	1.000000	0.508475	0.674157	0.754237
KNN	constant	None	None	0.902811	0.844311	0.597458	0.699752	0.785845
KNN	knn	None	None	0.907631	0.862275	0.610169	0.714640	0.793687
KNN	median	None	None	0.897189	0.803371	0.605932	0.690821	0.785622

Figure 12:Imputation results



Evaluation of Oversampling Methods

Different oversampling methods such as SMOTE, Borderline SMOTE,ADASYN,SMOTEENC are used in our experiment. The performance of different oversampling methods is compared with KNN imputation. Test data is used for evaluation of the model as shown in figure9. We can observe that with oversampling the models gave us better f1 results.



Model	Imputation Method	Oversampling Method	Dimension reduction	Accuracy	Precision	Recall	F1-Score	ROC-AUC Score
Logistic Regression	knn	smote	None	0.718072	0.360775	0.631356	0.459168	0.684855
Logistic Regression	knn	borderline	None	0.662651	0.304255	0.605932	0.405099	0.640924
Logistic Regression	knn	adasyn	None	0.637751	0.276507	0.563559	0.370990	0.609332
Logistic Regression	knn	smotenc	None	0.735743	0.381074	0.631356	0.475279	0.695757
SVM	knn	smote	None	0.918072	0.793860	0.766949	0.780172	0.860184
SVM	knn	borderline	None	0.915663	0.781116	0.771186	0.776119	0.860321
SVM	knn	adasyn	None	0.913253	0.773504	0.766949	0.770213	0.857211
SVM	knn	smotenc	None	0.905221	0.737903	0.775424	0.756198	0.855502
DT	knn	smote	None	0.867470	0.625442	0.750000	0.682081	0.822473
DT	knn	borderline	None	0.811245	0.501441	0.737288	0.596913	0.782916
DT	knn	adasyn	None	0.805622	0.490964	0.690678	0.573944	0.761593
DT	knn	smotenc	None	0.834538	0.550336	0.694915	0.614232	0.781055
Random Forest	knn	smote	None	0.922088	0.835749	0.733051	0.781038	0.849677
Random Forest	knn	borderline	None	0.906827	0.747934	0.766949	0.757322	0.853247
Random Forest	knn	adasyn	None	0.911647	0.756098	0.788136	0.771784	0.864335
Random Forest	knn	smotenc	None	0.914056	0.794521	0.737288	0.764835	0.846345
KNN	knn	smote	None	0.828112	0.530055	0.822034	0.644518	0.825784
KNN	knn	borderline	None	0.825703	0.525745	0.822034	0.641322	0.824297
KNN	knn	adasyn	None	0.800803	0.485646	0.860169	0.620795	0.823544
KNN	knn	smotenc	None	0.861044	0.613718	0.720339	0.662768	0.807147

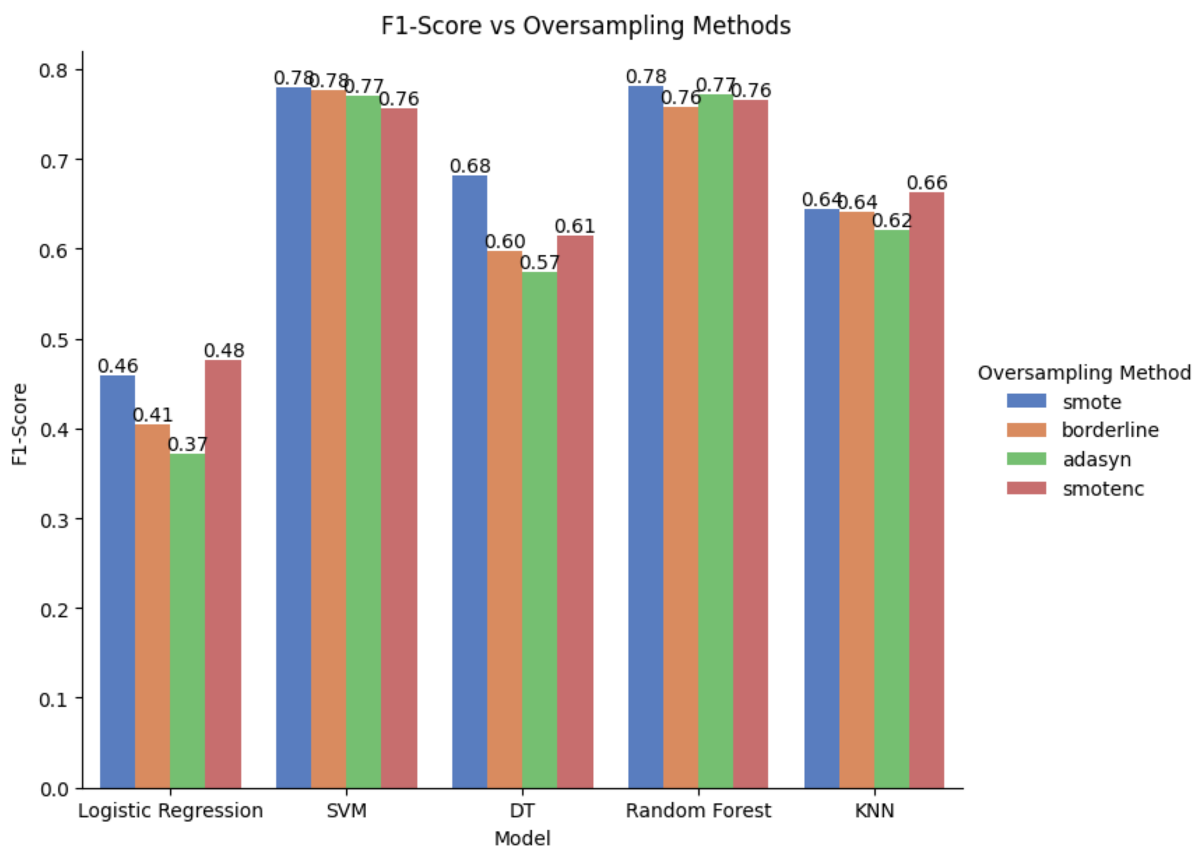
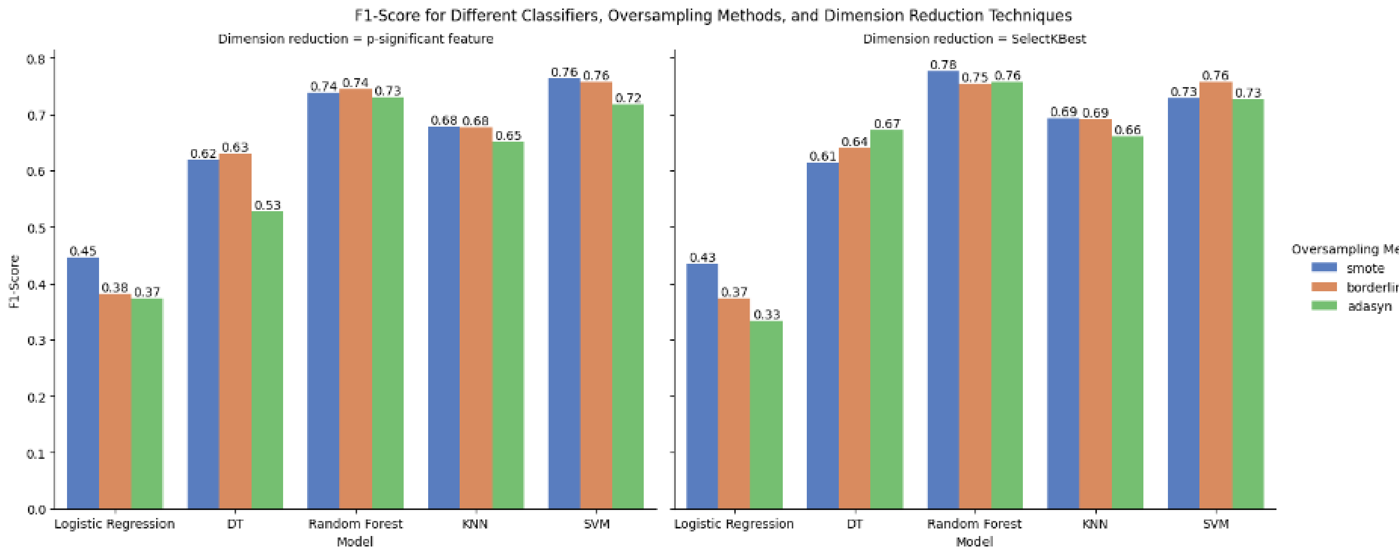
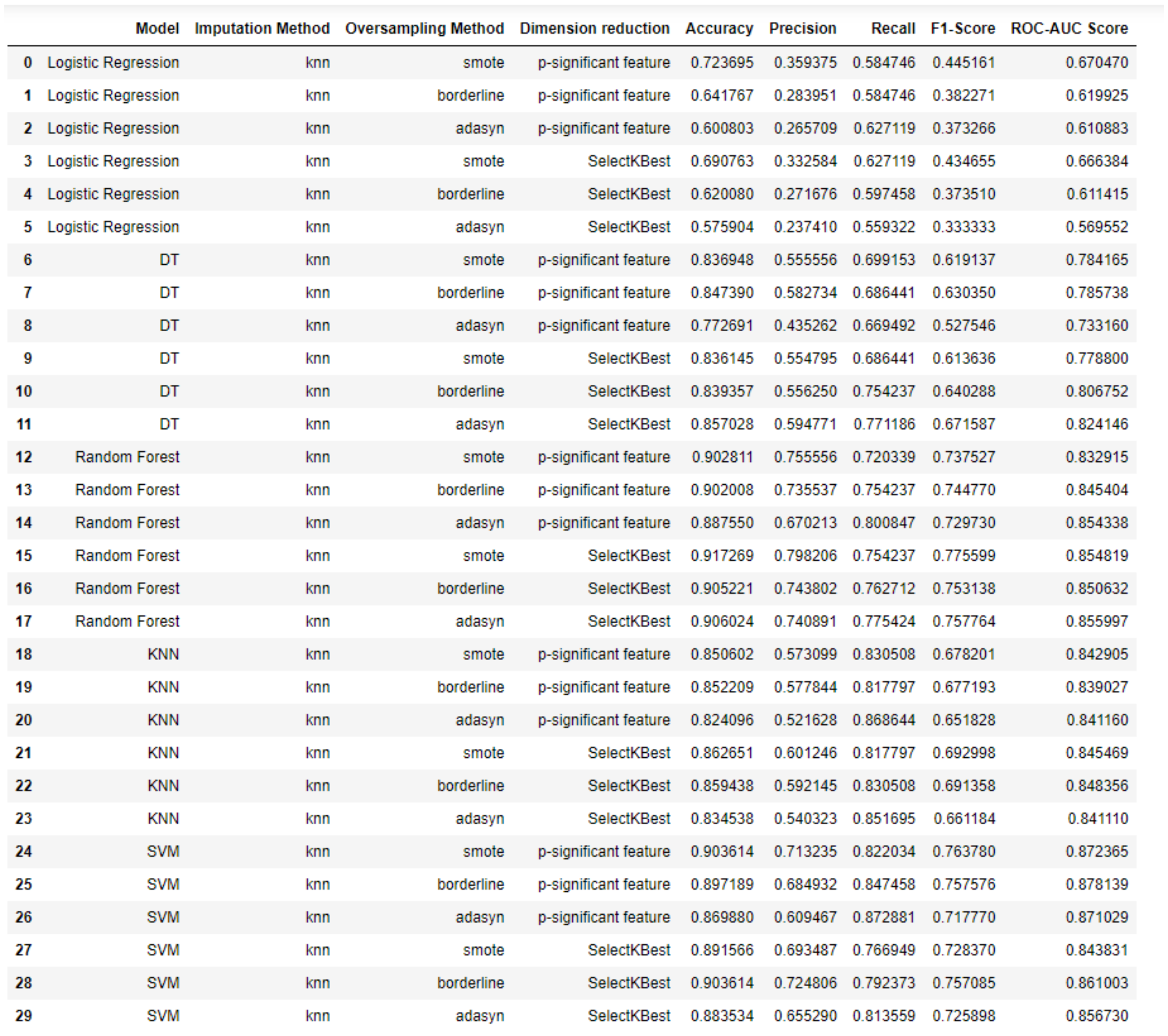


Figure 13:Evaluation of Oversampling Methods with KNN Imputation

Evaluation of Dimension reduction Techniques with different oversampling techniques:

Different dimensionality techniques like getting significant features with p value<0.05, Select K best features were performed for different oversampling methods such as SMOTE, Borderline SMOTE,ADASYN in the experiment. The performance of different dimension reduction techniques with oversampling methods are compared with KNN imputation. Test data is used for evaluation of the model as shown in figure10. Getting significant features with p value<0.05, Select K best features was produced better results than other techniques like kernel PCA.





	Model	Imputation Method	Oversampling Method	Dimension reduction	Accuracy	Precision	Recall	F1-Score	ROC-AUC Score
0	Logistic Regression	knn	smote	p-significant feature	0.723695	0.359375	0.584746	0.445161	0.670470
1	Logistic Regression	knn	borderline	p-significant feature	0.641767	0.283951	0.584746	0.382271	0.619925
2	Logistic Regression	knn	adasyn	p-significant feature	0.600803	0.265709	0.627119	0.373266	0.610883
3	Logistic Regression	knn	smote	SelectKBest	0.690763	0.332584	0.627119	0.434655	0.666384
4	Logistic Regression	knn	borderline	SelectKBest	0.620080	0.271676	0.597458	0.373510	0.611415
5	Logistic Regression	knn	adasyn	SelectKBest	0.575904	0.237410	0.559322	0.333333	0.569552
6	DT	knn	smote	p-significant feature	0.836948	0.555556	0.699153	0.619137	0.784165
7	DT	knn	borderline	p-significant feature	0.847390	0.582734	0.686441	0.630350	0.785738
8	DT	knn	adasyn	p-significant feature	0.772691	0.435262	0.669492	0.527546	0.733160
9	DT	knn	smote	SelectKBest	0.836145	0.554795	0.686441	0.613636	0.778800
10	DT	knn	borderline	SelectKBest	0.839357	0.556250	0.754237	0.640288	0.806752
11	DT	knn	adasyn	SelectKBest	0.857028	0.594771	0.771186	0.671587	0.824146
12	Random Forest	knn	smote	p-significant feature	0.902811	0.755556	0.720339	0.737527	0.832915
13	Random Forest	knn	borderline	p-significant feature	0.902008	0.735537	0.754237	0.744770	0.845404
14	Random Forest	knn	adasyn	p-significant feature	0.887550	0.670213	0.800847	0.729730	0.854338
15	Random Forest	knn	smote	SelectKBest	0.917269	0.798206	0.754237	0.775599	0.854819
16	Random Forest	knn	borderline	SelectKBest	0.905221	0.743802	0.762712	0.753138	0.850632
17	Random Forest	knn	adasyn	SelectKBest	0.906024	0.740891	0.775424	0.757764	0.855997
18	KNN	knn	smote	p-significant feature	0.850602	0.573099	0.830508	0.678201	0.842905
19	KNN	knn	borderline	p-significant feature	0.852209	0.577844	0.817797	0.677193	0.839027
20	KNN	knn	adasyn	p-significant feature	0.824096	0.521628	0.868644	0.651828	0.841160
21	KNN	knn	smote	SelectKBest	0.862651	0.601246	0.817797	0.692998	0.845469
22	KNN	knn	borderline	SelectKBest	0.859438	0.592145	0.830508	0.691358	0.848356
23	KNN	knn	adasyn	SelectKBest	0.834538	0.540323	0.851695	0.661184	0.841110
24	SVM	knn	smote	p-significant feature	0.903614	0.713235	0.822034	0.763780	0.872365
25	SVM	knn	borderline	p-significant feature	0.897189	0.684932	0.847458	0.757576	0.878139
26	SVM	knn	adasyn	p-significant feature	0.869880	0.609467	0.872881	0.717770	0.871029
27	SVM	knn	smote	SelectKBest	0.891566	0.693487	0.766949	0.728370	0.843831
28	SVM	knn	borderline	SelectKBest	0.903614	0.724806	0.792373	0.757085	0.861003
29	SVM	knn	adasyn	SelectKBest	0.883534	0.655290	0.813559	0.725898	0.856730

Figure 14:Evaluation of feature selection techniques with oversampling

Conclusion: SVM and Random Forest performed better

References

[1] Aktar, M. W., Sengupta, D., & Chowdhury, A. (2009). Impact of pesticides use in agriculture: their benefits and hazards. *Interdisciplinary toxicology*, *2*(1), 1–12. https://doi.org/10.2478/v10102-009-0001-7

[2] WHO/FAO (2014) International Code of Conduct on pesticide management. Food and Agriculture Organization of the United Nations/World Health Organization, Rome/Geneva

[3] Sharma, A. K., Gaur, K., Tiwari, R. K., & Gaur, M. S. (2011). Computational interaction analysis of organophosphorus pesticides with different metabolic proteins in humans. *Journal of biomedical research*, *25*(5), 335–347. https://doi.org/10.1016/S1674-8301(11)60045-6

[4] Mittal, S., Kaur, G., & Vishwakarma, G. S. (2014). Effects of environmental pesticides on the health of rural communities in the Malwa Region of Punjab, India: *A review. Human and Ecological Risk Assessment,* 20, 366–387.

[5] Wang, X., Yu, D., Ma, L., Lu, X., Song, J., & Lei, M. (2022). Using big data searching and machine learning to predict human health risk probability from pesticide site soils in China. *Journal of environmental management*, *320,* 115798. https://doi.org/10.1016/j.jenvman.2022.115798

[6] Ehsan Elahi, Cui Weijun, Huiming Zhang, Majid Nazeer, Agricultural intensification and damages to human health in relation to agrochemicals: *Application of artificial intelligence, Land Use Policy,* Volume 83, 2019, Pages 461-474, ISSN 0264-8377. https://doi.org/10.1016/j.landusepol.2019.02.023.

[7] Kim, Yeongmin, Minsu Chae, Namjun Cho, Hyowook Gil, and Hwamin Lee. 2022. "Machine Learning-Based Prediction Models of Acute Respiratory Failure in Patients with Acute Pesticide Poisoning" *Mathematics* 10, no. 24: 4633. https://doi.org/10.3390/math10244633

[8] Wan, E. T., Darssan, D., Karatela, S., Reid, S. A., & Osborne, N. J. (2021). Association of Pesticides and Kidney Function among Adults in the US Population 2001-2010. *International journal of environmental research and public health*, *18*(19), 10249. https://doi.org/10.3390/ijerph181910249

[9] Sarita Limbu , Cyril Zakka & Sivanesan Dakshanamurthy. (2022). Predicting Environmental Chemical Toxicity using a New Hybrid Deep Machine Learning Method. https://www.mdpi.com/2305-6304/10/11/706

[10] Ahmet Murat Erturan , Gül Karaduman , Habibe Durmaz. (2023). Machine learning-based approach for efficient prediction of toxicity of chemical gases using feature selection. *Journal of Hazardous Materials volume 455, 5 August 2023, 131616.* https://doi.org/10.1016/j.jhazmat.2023.1316