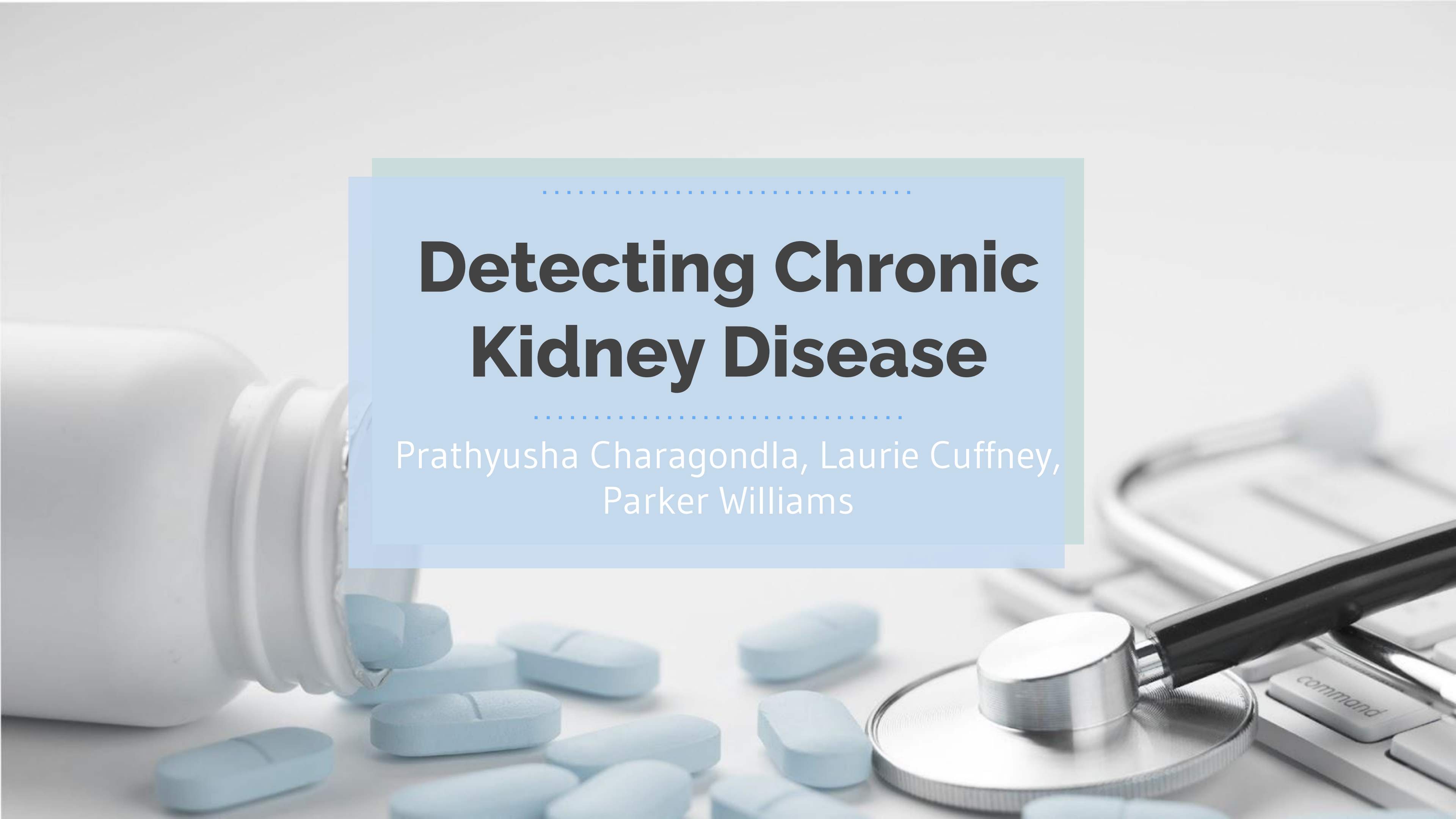
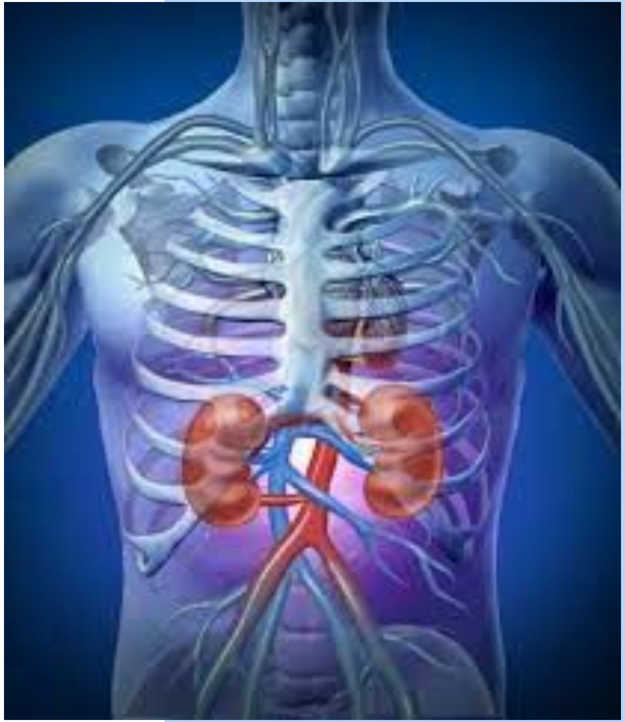


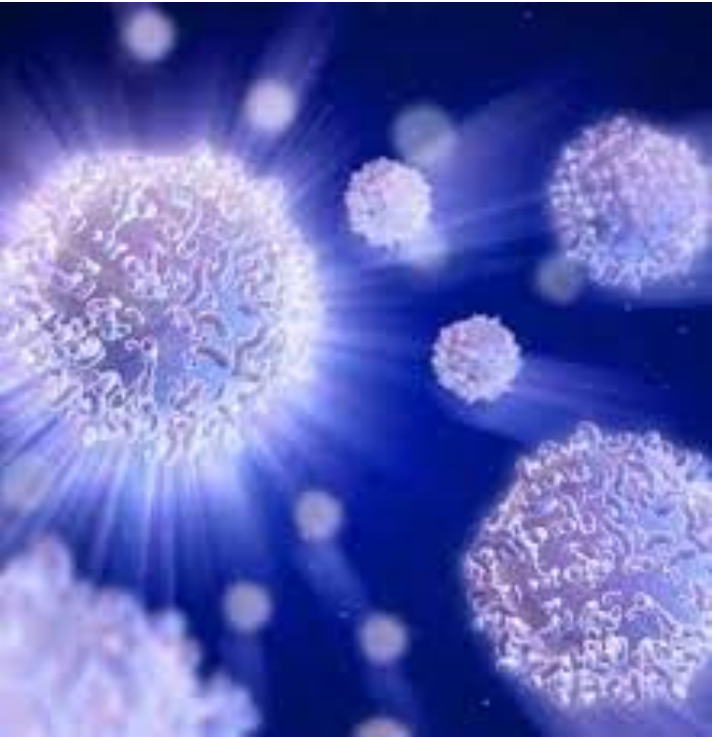
Detecting Chronic Kidney Disease

Prathyusha Charagondla, Laurie Cuffney,
Parker Williams

Chronic Kidney Disease (CKD) 'The Silent Disease'



- A diagnosis of kidney disease means that a person's kidneys are damaged and cannot filter blood the way they should.
- Each year, kidney disease kills more people than breast or prostate cancer. In 2013, more than 47,000 Americans died from kidney disease.[1]
- There are 5 stages of kidney disease, but in its early stages and can go undetected until it is very advanced.
- The overall prevalence of CKD in the general population is approximately 14 percent.
- $50 Billion in Medicare was spent on CKD in 2013



The Data

- From UC Irvine Machine Learning Repository.
- Covers a two month period in India
- There 400 observations, 25 features, and a classification variable (Y/N CKD)
- Classification variable: 250 observations CKD=Y and 150 rows CKD=N
- The data includes numeric, binary, and categorical features
- There are just over 1,000 null values across all 25 features.
- The white blood cell count and red blood count variables have the most nulls values with 105 and 130 observations respectively.

Data Preprocessing: Cleaning

Remove Extraneous Tabs

- Several columns read in with extra tab characters
- Causes issues with enumeration

```
labels = data.classification.unique()
labels
```

```
array(['ckd', 'ckd\t', 'notckd'], dtype=object)
```

Correct Data Types

- Three potential numeric feature columns read in as type object
- rc, wc, pcv

```
pcv                     object
wc                      object
rc                      object
```

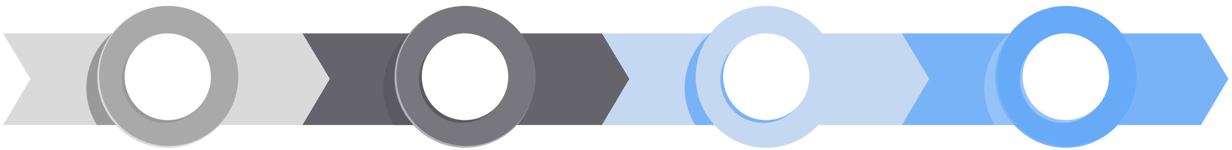
pcv	wc	rc
44	7800	5.2
38	6000	NaN
31	7500	NaN
32	6700	3.9
35	7300	4.6

Additional Data Preprocessing

Binarize Categorical Variables

- 5 yes/no
- 2 abnormal/normal
- 2 present/notpresent
- 1 good/poor

Select features and remove missing values



Recode Class labels

1: Chronic Kidney Disease
0: Not Chronic Kidney Disease

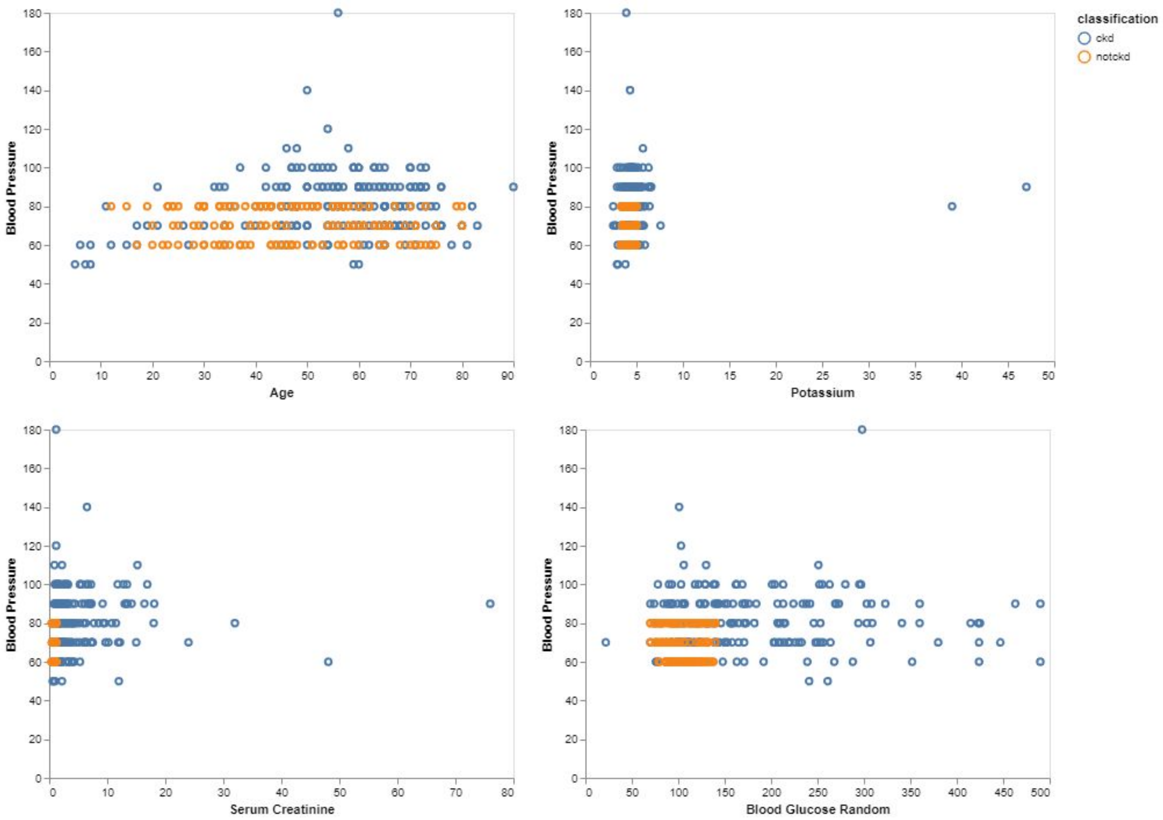
Standardize features

Using sklearn preprocessing.scale( )

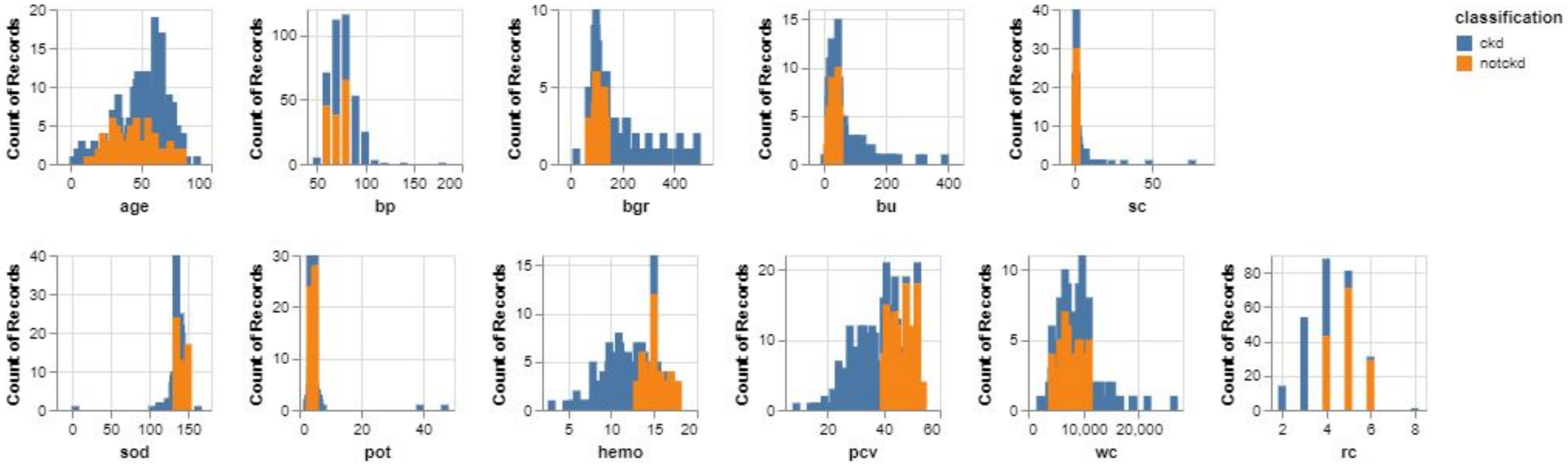
EDA

- Blood pressure vs
  - Age,
  - Potassium
  - Serum Creatinine
  - Blood Glucose Random

- Heavy overlap in between the two classification labels

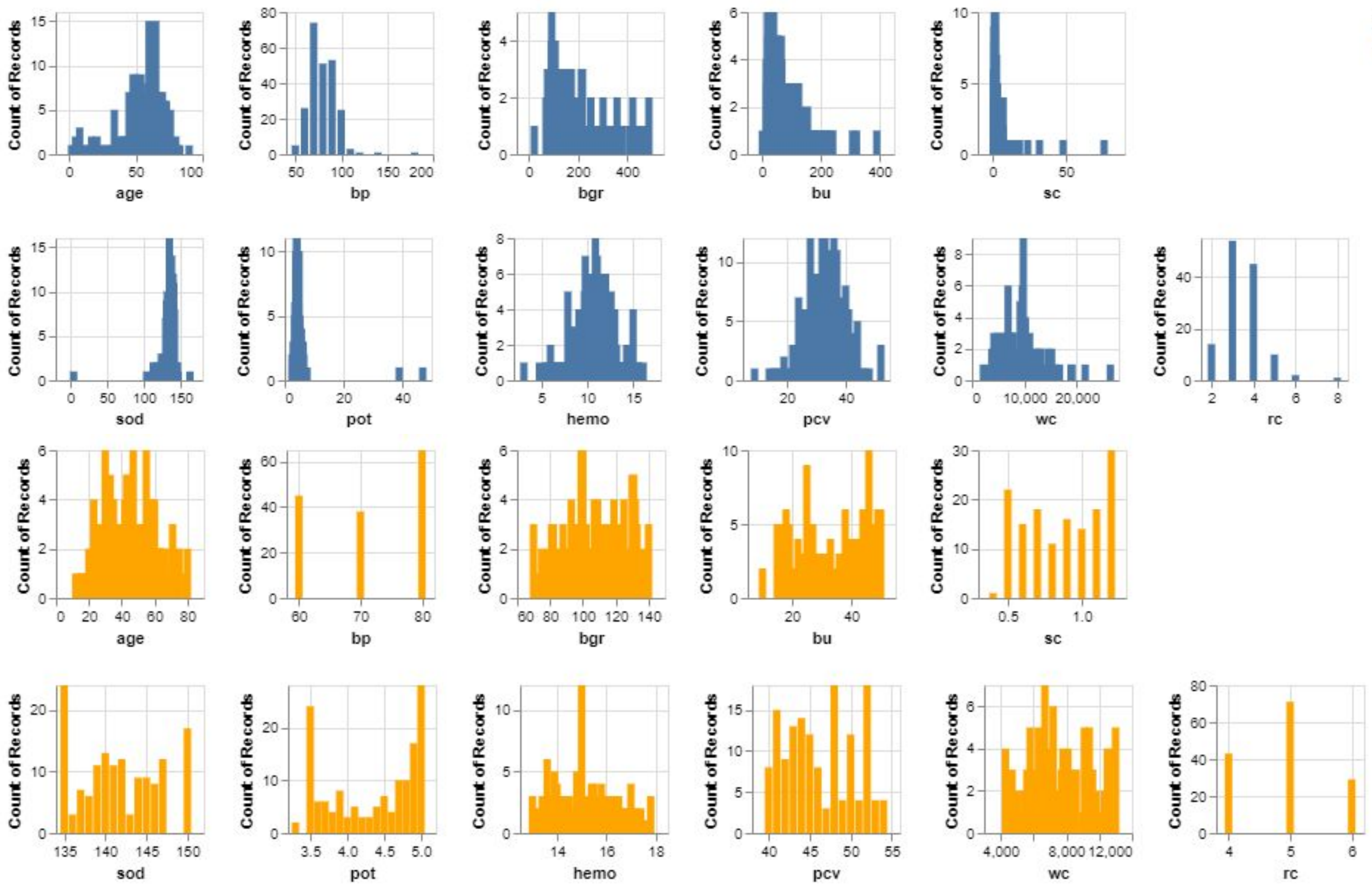


Numeric Feature Variable Distributions



Chronic Kidney Disease

Not Chronic Kidney Disease



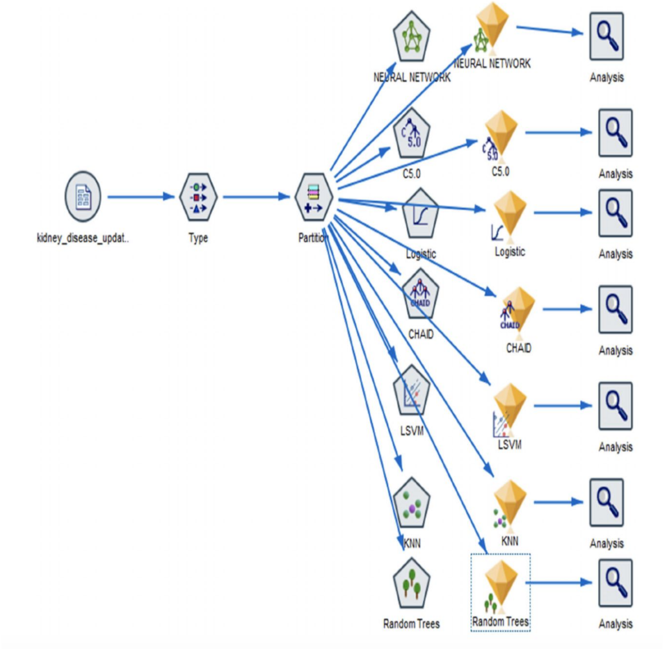


Table 3: Different Algorithm with different number of attribute

Algorithm	Accuracy (25 attributes)	Accuracy (20 attributes)	Accuracy (15 attributes)	Accuracy (10 attributes)
Naïve Bayes	95%	95.5%	96%	95.25%
SVM	97.75%	98.25%	98.5%	97.75%
Decision tree	99%	99%	99%	98%
KNN	95.75%	96	97.5%	96.25%

Literature

- Previously analyzed in multiple papers
- Used to predict/classify instances of Chronic Kidney Disease based on a set of features
- Known models used include: Logistic Regression, KNN, Decision Tree, Naive Bayes, SVM
- On average, both papers found that SVMs performed well (~97% Accuracy between both papers)

1. Chittora, Pankaj & Sandeep, Chaurasia & Chakrabarti, Prasun & Kumawat, Gaurav & Chakrabarti, Tulika & Leonowicz, Zbigniew & Jasiński, Michał & Jasiński, Łukasz & Gono, Radomir & Jasińska, Elżbieta & Bolshev, Vadim. (2021). Prediction of Chronic Kidney Disease - A Machine Learning Perspective. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3053763.

2. Tazin, Nusrat & Sabab, Shahed & Chowdhury, Muhammed. (2016). Diagnosis of Chronic Kidney Disease using effective classification and feature selection technique. 1-6. 10.1109/MEDITEC.2016.7835365.

Modeling Components

Features

2 features

- **bp, sc**

16 features

- pc, rbc, htn, cad, ane, su, sg, age, bp, bgr, bu, sc, sod, pot, hemo, pcv

24 features

- All available features

	Number of Features	**2**	**16**	**24**
Train Data	Drop NAs	182	84	77
	Imputed	196	115	114
Dev Data	Drop NAs	79	36	33
	Imputed	84	50	49
Test Data	Drop NAs	112	52	48
	Imputed	120	71	70

Model: Support Vector Classification

kernel: rbf

C = 1.0, gamma = 5

	precision	recall	f1-score	support
0	1.00	0.68	0.81	28
1	0.00	0.00	0.00	0
accuracy			0.68	28
macro avg	0.50	0.34	0.40	28
weighted avg	1.00	0.68	0.81	28

- Initial model has moderately improved accuracy
- Only predicting 0 label (not ckd)

GridSearch CV

Tuning

C = [0.1, 1, 10, 100, 1000]

gamma = [1, 0.1, 0.01, 0.001, 0.0001]

kernel =['linear','rbf']

kernel: linear

C=0.1, gamma = 1

	precision	recall	f1-score	support
0	1.00	1.00	1.00	36
1	1.00	1.00	1.00	16
accuracy			1.00	52
macro avg	1.00	1.00	1.00	52
weighted avg	1.00	1.00	1.00	52

- Huge improvement in accuracy on dev data

Model: Support Vector Classification

kernel: linear

$C = 1.0, gamma = 1$

	precision	recall	f1-score	support
0	1.00	0.97	0.99	37
1	0.94	1.00	0.97	15
accuracy			0.98	52
macro avg	0.97	0.99	0.98	52
weighted avg	0.98	0.98	0.98	52

- 98% Accuracy on the test data
- Best SVM model on 16 features

Model: Bernoulli Naive Bayes

Dev Data Performance

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

- 100% accuracy of the dev data

Test Data Performance

	precision	recall	f1-score	support
0	1.00	0.96	0.98	25
1	0.91	1.00	0.95	10
accuracy			0.97	35
macro avg	0.95	0.98	0.97	35
weighted avg	0.97	0.97	0.97	35

- 97% accuracy on the test data, slightly below the SVM model

Model: Decision Trees

Dev Data
Depth = 5

Prediction Report for Dataset with No Nulls on Dev Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

- Models with 16 variables had around 100% accuracy in Dev and Test

Parameter Tuning

max_depth = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, None]

Test Data
Depth = 5

Prediction Report for Dataset with No Nulls on Test Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	24
1	1.00	1.00	1.00	11
accuracy			1.00	35
macro avg	1.00	1.00	1.00	35
weighted avg	1.00	1.00	1.00	35

Model: Ensemble Learning: Bagging

Dev Data
N_Estimators = 500

```
Prediction Report for Dataset with No Nulls on Dev Data
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        19
           1       1.00      1.00      1.00         9

    accuracy                           1.00        28
   macro avg       1.00      1.00      1.00        28
weighted avg       1.00      1.00      1.00        28
```

- Models with 16 variables had around 100% accuracy in Dev and 97% in Test

Parameter Tuning

n_estimators = [10, 50, 100, 500, 1000, 5000]

Test Data
N_Estimators = 500

```
Prediction Report for Dataset with No Nulls on Test Data
              precision    recall  f1-score   support

           0       0.96      1.00      0.98        24
           1       1.00      0.91      0.95        11

    accuracy                           0.97        35
   macro avg       0.98      0.95      0.97        35
weighted avg       0.97      0.97      0.97        35
```

Model: Ensemble Learning: AdaBoost

Dev Data

N_Estimators = 100

Prediction Report for Dataset with No Nulls on Dev Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

- Models with 16 variables had 100% accuracy in Dev and Test

Parameter Tuning

n_estimators = [10, 50, 100, 500, 1000, 5000]

Test Data

N_Estimators = 100

Prediction Report for Dataset with No Nulls on Test Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	24
1	1.00	1.00	1.00	11
accuracy			1.00	35
macro avg	1.00	1.00	1.00	35
weighted avg	1.00	1.00	1.00	35

Model: Random Forest

Dev Data

$criterion = entropy$

Prediction Report for Dataset with No Nulls on Dev Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

- Models with 16 variables had around 100% accuracy in Dev and Test

Test Data

$criterion = entropy$

Prediction Report for Dataset with No Nulls on Test Data

	precision	recall	f1-score	support
0	1.00	1.00	1.00	24
1	1.00	1.00	1.00	11
accuracy			1.00	35
macro avg	1.00	1.00	1.00	35
weighted avg	1.00	1.00	1.00	35

Model: KNN

Dev Data

N_Neighbors=1

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

GridSearch CV

Tuning

n_neighbors = [1,50,1]

Test Data

N_Neighbors=1

	precision	recall	f1-score	support
0	0.96	1.00	0.98	24
1	1.00	0.91	0.95	11
accuracy			0.97	35
macro avg	0.98	0.95	0.97	35
weighted avg	0.97	0.97	0.97	35

-

Model: Logistic Regression

Dev Data

$C=.16$

	precision	recall	f1-score	support
0	1.00	1.00	1.00	19
1	1.00	1.00	1.00	9
accuracy			1.00	28
macro avg	1.00	1.00	1.00	28
weighted avg	1.00	1.00	1.00	28

GridSearch CV

Tuning

C = [0.1,0.5,0.02]

penalty='l2'

Test Data

$C=.16$

	precision	recall	f1-score	support
0	0.92	1.00	0.96	24
1	1.00	0.82	0.90	11
accuracy			0.94	35
macro avg	0.96	0.91	0.93	35
weighted avg	0.95	0.94	0.94	35

Model Accuracy Overview

	2 features		16 features		24 features	
	Raw	Imputed	Raw	Imputed	Raw	Imputed
SVM	65%	73%	98%	97%	100%	100%
Naive Bayes	61%	62%	97%	94%	100%	100%
Knn	72%	80%	97%	94%	100%	94%
Logistic	63%	73%	94%	96%	100%	100%
Decision Trees	72%	80%	100%	92%	97%	98%
Random Forest	72%	80%	100%	98%	100%	98%
Bagging	72%	80%	97%	94%	97%	98%
AdaBoost	72%	80%	100%	94%	97%	98%

Conclusion

- Multiple models (KNN, Logistic Regression, SVM, Naive Bayes) had perfect accuracy when we included all features
  - We believe this is due to our limited sample size and lack of variation.
  - This is causing over-fitting which we are benefiting from given our small test holdouts.
  - Accuracy decreases in some of our models when we impute and increase sample size.
- Mean imputation was not beneficial except when using a small number of features.

Thanks!

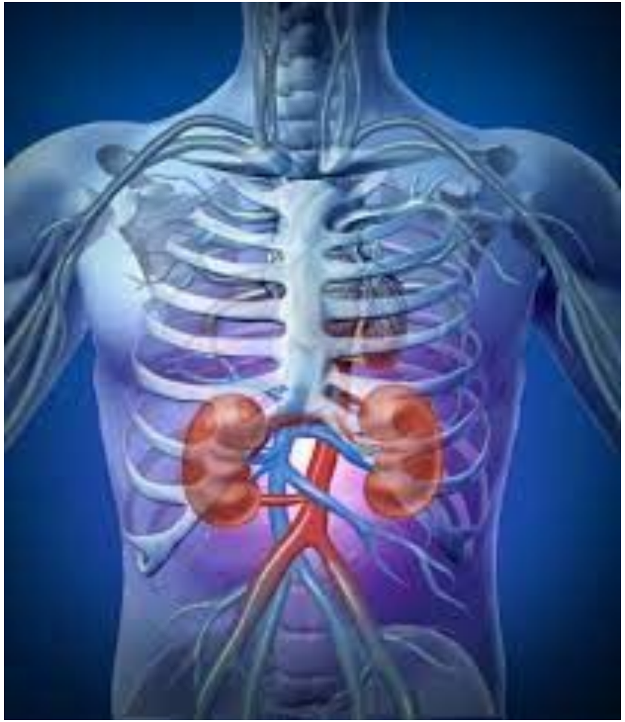
Does anyone have any questions?

Feel free to share feedback via slack.

Original Unformatted Slides

Chronic Kidney Disease (CKD) -- 'The Silent Disease'

- A diagnosis of kidney disease means that a person's kidneys are damaged and cannot filter blood the way they should.
- Each year, kidney disease kills more people than breast or prostate cancer. In 2013, more than 47,000 Americans died from kidney disease.[1]
- There are 5 stages of kidney disease, but in its early stages and can go undetected until it is very advanced.
- The overall prevalence of CKD in the general population is approximately 14 percent.
- $50 Billion in Medicare was spend on CKD in 2013



Literature

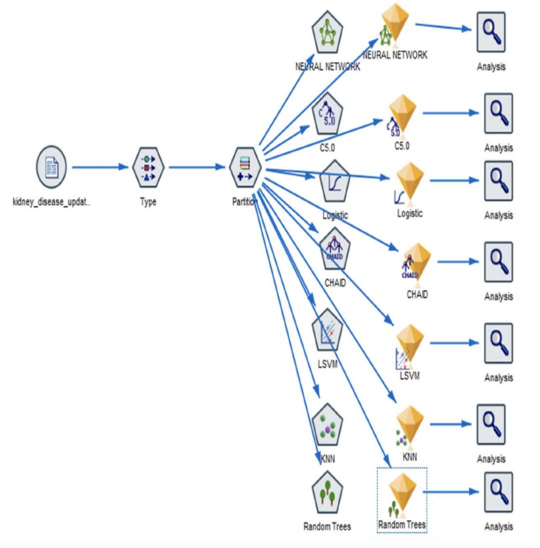


Table 3: Different Algorithm with different number of attribute

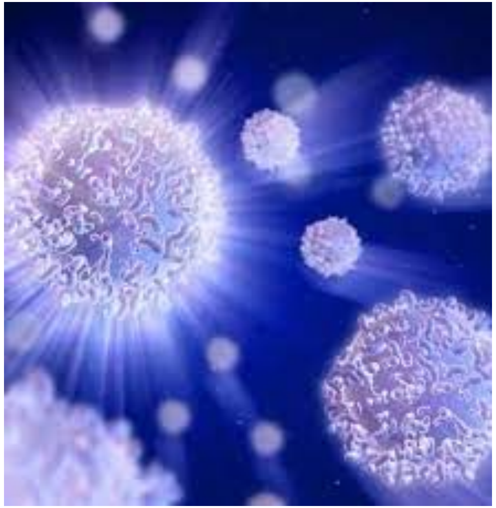
Algorithm	Accuracy (25 attributes)	Accuracy (20 attributes)	Accuracy (15 attributes)	Accuracy (10 attributes)
Naïve Bayes	95%	95.5%	96%	95.25%
SVM	97.75%	98.25%	98.5%	97.75%
Decision tree	99%	99%	99%	98%
KNN	95.75%	96	97.5%	96.25%

- Previously analyzed in multiple papers
- Used to Predict/Classify instances of Chronic Kidney Disease
- Known models used include: Logistic Regression, KNN, Decision Tree, Naive Bayes, SVM
- On average, both papers found that SVMs performed will post feature-selection (~97% Accuracy between both papers)

1. Chittora, Pankaj & Sandeep, Chaurasia & Chakrabarti, Prasun & Kumawat, Gaurav & Chakrabarti, Tulika & Leonowicz, Zbigniew & Jasiński, Michał & Jasiński, Łukasz & Gono, Radomir & Jasińska, Elżbieta & Bolshev, Vadim. (2021). Prediction of Chronic Kidney Disease - A Machine Learning Perspective. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3053763.

2. Tazin, Nusrat & Sabab, Shahed & Chowdhury, Muhammed. (2016). Diagnosis of Chronic Kidney Disease using effective classification and feature selection technique. 1-6. 10.1109/MEDITEC.2016.7835365.

The Data



- From UC Irvine Machine Learning Repository.
- Covers a two month period in India
- There 400 observations, 25 features, and a classification variable (Y/N CKD)
- Classification variable: 250 observations CKD=Y and 150 rows CKD=N
- The data include numeric, binary, and categorical features
- There are just over 1,000 null values across all 25 features.
- The white blood cell count and red blood count variables have the most nulls values with 105 and 130 observations respectively.