A MACHINE LEARNING METHODOLOGY FOR DIAGNOSING CHRONIC KIDNEY DISEASE

MURALI PONAGANTI
Assistant Professor
Department Of MCA
Sree Chaitanya College of Engineering,Karimnagar

ABSTRACT

Chronic kidney disease (CKD) is a global health problem with high morbidity and mortality rate, and it induces other diseases. Since there are no obvious symptoms during the early stages of CKD, patients often fail to notice the disease. Early detection of CKD enables patients to receive timely treatment to ameliorate the progression of this disease. Machine learning models can effectively aid clinicians achieve this goal due to their fast and accurate recognition performance. In this study, we propose a machine learning methodology for diagnosing CKD. The CKD data set was obtained from the University of California Irvine (UCI) machine learning repository, which has a large number of missing values. KNN imputation was used to fill in the missing values, which selects several complete samples with the most similar measurements to process the missing data for each incomplete sample. Missing values are usually seen in real-life medical situations because patients may miss some measurements for various reasons. After effectively filling out the incomplete data set, six machine learning algorithms (logistic regression, random forest, support vector machine, k-nearest neighbor, naive Bayes classifier and feed forward neural network) were used to establish models. Among these machine learning models, random forest achieved the best performance with 99.75% diagnosis accuracy. By analyzing the misjudgments generated by the established models, we proposed an integrated model that combines logistic regression and random forest by using perceptron, which could achieve an average accuracy of 99.83% after ten times of simulation. Hence, we speculated that this methodology could be applicable to more complicated clinical data for disease diagnosis.

Keyword: Chronic Kidney Disease, Machine Learning, KNN Imputation, Integrated Model

1.INTRODUCTION

Chronic kidney disease (CKD) is one of the most common diseases worldwide, affecting about 10% of the world's population [1]. The presence of kidney damage or impairment of kidney function are indications of chronic kidney disease [2], also characterized by decreased excretory renal work or proteinuria for more than 3 months [3]. Kidneys have millions of tiny blood vessels that act as filters to remove waste products from the blood. In some cases, this filtering system breaks down and the kidneys lose their ability to filter out waste products, leading to kidney disease. There is no single underlying cause of CKD, but worsening is generally irreversible and can lead to serious health problems [4]. In addition to the severity of chronic kidney disease, it makes the patient more susceptible to other diseases such as acute renal fever, poor oral absorption, and diarrhea [5].CKD in the early stages can be treatable, but the late stages lead to kidney failure. CKD is dangerous and could be a lifethreatening disease, killing 753 million people in 2016

[6]. Diabetes mellitus is considered the main cause of chronic kidney disease, but it is not the only one; in other cases, plant and environmental toxins can be the cause [4]. The main challenge in CKD isthe symptomsthat do not appear in the early stages, which can result in a 25% loss of kidney function due to late diagnosis . The best way to prevent CKD from progressing to renal failure is early diagnosis [6]. Most developing countries suffer from a shortage of specialized doctors, in addition to high diagnostic costs, especially in remote provinces. Therefore, the need to develop new technologies for diagnosing chronic kidney disease, which would help the doctor in the early detection of the disease, increased. Deep learning and machine learning (AI) play an important role in the medical field, especially in early detection and disease prediction. The most widely used technologies are ANN and SVM algorithms. These technologies have great advantages in many areas in the identification process as well as diagnosis [7,9]. The functionality of applying the kNN, ANN, and SVM methods in this research is to predict ifsomeone is at risk of developing CKD [10]. “Health is the greatest of all possessions; a pale cobbler is better than a sick king” is a proverb stated by Mr. Isaac Bickerstaff. Health is wealth, health is more important than anything else in the world. Human health can be affected by various diseases. Among themkidney disease plays a vital role. 10% of the population worldwide is affected by Chronic Kidney Disease (CKD) and over 2 million people all around the world are receiving treatment for dialysis or kidney transplantation, where dialysis or transplantation of kidney, is the final treatment given to a kidney patient, to survive. Millions of people die each year because of kidney disease [1].According to the Global Burden of Disease (GBD), in high income countries, they spent 2-4 % of their annual healthcare budget to treat people with, end stage kidney failure [2]. GBD also states that by the end of year 2030, the number of people getting kidney dialysis treatment may increase up to 4 million. In India, the actual number of people with kidney disease is unknown, since there is no renal register. “The number of people undergoing dialysis is increasing by 10- 15 % every year and no to much importance is given to kidney disorders as it is still under-the-radar condition", said Sudeep Singh Sachdev, a consultant for Nephrology at Max Super Speciality Hospital [3].Health minister of India states that, two thousand new dialysis centres will be established in the district level hospitals all around India by the end of year 2018. For kidney disease, over 60% of the people do not get proper medical treatment on time, which may cause death. Only 4% of the people with CKD get a donor for kidney transplantation. All these statistics predict that, the number of people affected by kidney diseases is increasing every year. To prevent this, if CKD is detected in the early stage, it is possible to slow down or stop the progression of kidney disease and save people from ending up with dialysis or transplantation to live. To know about the early detection of CKD, first it is important to know about the anatomy of kidney, anatomy of nephron – the functional unit of kidney, degradation of nephron, the major classification of kidney disease, CKD, types of CKD, the symptoms of CKD, different stages of CKD and progression of kidney disease, Detailed discussion on thee can be found in the later sections.

Conventional techniques to detect CKD are based on urine test, blood test, and biopsy. It is not possible to detect the CKD in early stage and hence medical imaging is used. Medical imaging is the technique of creating visual representations of the interior parts of the body for clinical analysis and medical intervention. Imaging technologies used to create the visual representation of the internal organs of a human body are X-ray radiography, magnetic resonance imaging(MRI), Ultrasonography, Endoscope, Elastography, Tactile imaging, Computer Tomography (CT), Thermography, Medical Photography, Positron Emission Tomography (PET) and Single – Photon Emission Computer Tomography (SPECT). Among these imaging techniques, to visualise the kidney for a CKD patient, Ultrasonography is the most suitable one, since it won't worsen the function of kidney further. The detailed description of ultrasound image, working principle of ultrasound scanners, transducers, various ultra sound display modes, various ultrasound imaging techniques are all discussed inthis chapter. Though ultrasound is the most widely used technique, still there are issues in getting critical details of the progression of the disease. Digital imaging is a viable solution to solve this problem. The various image processing modules used for early detection of CKD are the image enhancement, image segmentation, image feature extraction and analysis. To enhance the digital quality of ultrasound image, the image enhancement techniques are used. Image segmentation is used to separate the kidney from the ultrasound image. After segmenting thekidney, extracting the features that are helpful for the doctors to diagnose the progression of CKD in the early stage are discussed.

2.SYSTEM ANALYSIS

EXISTING SYSTEM:

KNN imputation is used to fill in the missing values. To our knowledge, this is the first time that KNN imputation has been used for the diagnosis of CKD. In addition, building an integrated model is also a good way to improve the performance of separate individual models. The proposed methodology might effectively deal with the scene where patients are missing certain measurements before being diagnosed. In addition, the resulting integrated model shows a higher accuracy. Therefore, it is speculated that this methodology might be applicable to the clinical data in the actual medical diagnosis[5].

PROPOSED SYSTEM:

The classifiers were first established by different machine learning algorithms to diagnose the data samples. Among these models, those with better performance were selected as potential components. By analysing their misjudgements, the component models were determined. An integrated model was then established to achieve higher performance[6].

3. Methodology

This is the heart of your paper. Explain the machine learning algorithms and techniques used for CKD diagnosis. Commonly used techniques include[8]:

- Logistic Regression
- Decision Trees
- Random Forests
- Support Vector Machines
- Neural Networks
- Gradient Boosting

Discuss how you've adapted or customized these algorithms for CKD diagnosis.

Explain the training process, hyper parameter tuning, and any special considerations.

Data Collection:

Explain the sources of data used in your study. This might include patient records, medical databases, or clinical trials. Discuss the data pre-processing steps, such as handling missing values, outlier detection, and feature selection/engineering. Address any challenges specific to CKD data.

Feature

Describe the features used in your machine learning model. These could include medical measurements, patient demographics, lab results, etc. Explain any domain-specific insights that led to the selection of these features.

PROPOSED MODEL:

In this section, the classifiers were first established by d-different machine learning algorithms to diagnose the data samples. Among these models, those with better performance were selected as potential components. By analysing theirmisjudgments, the component models were determined. An integrated model was then established to achieve higher performance

4. RESULTS

Dataset loading

```
df = pd.read_csv("kidney.csv")
```

```
df.head()
```

	id	age	bp	sg	al	su	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wc	rc	htn	dm	cad	appet	pe	ane
0	0	48.0	80.0	1.020	1.0	0.0	NaN	normal	notpresent	notpresent	121.0	36.0	1.2	NaN	NaN	15.4	44	7800	5.2	yes	yes	no	good	no	no
1	1	7.0	50.0	1.020	4.0	0.0	NaN	normal	notpresent	notpresent	NaN	18.0	0.8	NaN	NaN	11.3	38	6000	NaN	no	no	no	good	no	no
2	2	62.0	80.0	1.010	2.0	3.0	normal	normal	notpresent	notpresent	423.0	53.0	1.8	NaN	NaN	9.6	31	7500	NaN	no	yes	no	poor	no	yes
3	3	48.0	70.0	1.005	4.0	0.0	normal	abnormal	present	notpresent	117.0	56.0	3.8	111.0	2.5	11.2	32	6700	3.9	yes	no	no	poor	yes	yes
4	4	51.0	80.0	1.010	2.0	0.0	normal	normal	notpresent	notpresent	106.0	26.0	1.4	NaN	NaN	11.6	35	7300	4.6	no	no	no	good	no	no

Data Analysis:

Blood Pressure (bp)

```
df.bp.unique()
```

```
array([ 80.,  50.,  70.,  90.,  nan, 100.,  60., 110., 140., 180., 120.])
```

```
df.bp.mode()[0]
```

```
80.0
```

```
df.bp = df.bp.replace(np.NaN, df.bp.mode()[0])
```

Specific Gravity (sg)

```
df.sg.unique()
```

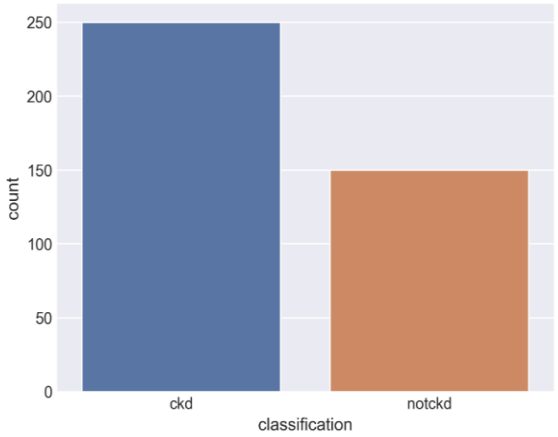
```
array([1.02 , 1.01 , 1.005, 1.015,  nan, 1.025])
```

```
df.sg.mode()[0]
```

```
1.02
```

```
df.sg = df.sg.replace(np.NaN, df.sg.mode()[0])
```

Classification



Logistic Regression algorithm accuracy

Over sample Logistic

```
lr = LogisticRegression(C=1, penalty='l2', solver='newton-cg')
lr.fit(X_train,y_train)

y_pred_lr = lr.predict(X_test)

print(metrics.classification_report(y_test, y_pred_lr))

lr_score = lr.score(X_train,y_train)
print(lr_score)

lr_score = lr.score(X_test,y_test)
print(lr_score)
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        44
           1       1.00      0.99      0.99        76

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120

0.9785714285714285
0.9916666666666667
```

Decision Tree Classifier

```
def dtree_grid_search(X, y):
    #create a dictionary of all values we want to test
    param_grid = { 'criterion':['gini','entropy'],'max_depth': np.arange(2, 15)}
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    # decision tree model
    dtree = DecisionTreeClassifier()
    #use gridsearch to test all values
    dtree_gscv = GridSearchCV(dtree, param_grid, cv=cv, n_jobs=-1, scoring='accuracy')
    #fit model to data
    dtree_gscv.fit(X, y)
    return dtree_gscv.best_params_
```

Over Sample Decision Tree

```
dTree = DecisionTreeClassifier(criterion = 'entropy', max_depth = 11)
dTree.fit(X_train, y_train)
print(dTree.score(X_train,y_train))
print(dTree.score(X_test,y_test))
y_pred_dtree = dTree.predict(X_test)
print(metrics.classification_report(y_test, y_pred_dtree))
```

```
1.0
0.9916666666666667
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        44
           1       1.00      0.99      0.99        76

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120
```

AdaBoosting

```
def ada_grid_search(X, y):
    #create a dictionary of all values we want to test
    param_grid = {'n_estimators':[10, 50, 100, 500], 'learning_rate': [0.0001, 0.001, 0.01, 0.1, 1.0]}
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    # AdaBoost model
    ada = AdaBoostClassifier()

    # Use gridsearch to test all values
    ada_gscv = GridSearchCV(ada, param_grid, n_jobs=-1, cv=cv, scoring='accuracy')
    #fit model to data
    grid_result = ada_gscv.fit(X, y)

    return ada_gscv.best_params_
```

Over Sample AdaBoost

```
abc1 = AdaBoostClassifier(n_estimators=500, learning_rate = 0.1)
abc1 = abc1.fit(X_train, y_train)

y_pred_abc1 = abc1.predict(X_test)

print(abc1.score(X_train, y_train))
print(abc1.score(X_test,y_test))

print(metrics.classification_report(y_test, y_pred_abc1))
```

```
1.0
1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        44
           1       1.00      1.00      1.00        76

    accuracy                           1.00       120
   macro avg       1.00      1.00      1.00       120
weighted avg       1.00      1.00      1.00       120
```

Random forest classifier

```
def rf_grid_search(X, y):
    #create a dictionary of all values we want to test
    param_grid = {
    'n_estimators': [5,10,20,40,50,60,70,80,100],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [4,5,6,7,8],
    'criterion' :['gini', 'entropy']
    }
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    # Random Forest model
    rf = RandomForestClassifier()

    #use gridsearch to test all values
    rf_gscv = GridSearchCV(rf, param_grid, cv=cv, n_jobs=-1, scoring='accuracy')
    #fit model to data
    rf_gscv.fit(X, y)

    return rf_gscv.best_params_
```

Over Sample Random Forest

```
rfc1 = RandomForestClassifier(n_estimators=70, max_features='sqrt', max_depth=7, criterion='entropy')
rfc1 = rfc1.fit(X_train, y_train)

y_pred_rf = rfc1.predict(X_test)

print(rfc1.score(X_train,y_train))
print(rfc1.score(X_test,y_test))

print(metrics.classification_report(y_test, y_pred_rf))
```

```
1.0
1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        44
           1       1.00      1.00      1.00        76

    accuracy                           1.00       120
   macro avg       1.00      1.00      1.00       120
weighted avg       1.00      1.00      1.00       120
```

KNN

```
def knn_grid_search(X, y):
    #create a dictionary of all values we want to test
    k_range = list(range(1, 31))
    param_grid = dict(n_neighbors=k_range)
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    knn = KNeighborsClassifier()

    #use gridsearch to test all values
    knn_gscv = GridSearchCV(knn, param_grid, cv=cv, scoring='accuracy', n_jobs=-1)
    #fit model to data
    knn_gscv.fit(X, y)

    return knn_gscv.best_params_
```

Over Sample kNN

```
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)

y_pred_knn = knn.predict(X_test)

print(knn.score(X_train, y_train))
print(knn.score(X_test, y_test))

print(metrics.classification_report(y_test, y_pred_knn))
```

```
1.0
0.9333333333333333
              precision    recall  f1-score   support

           0       0.88      0.95      0.91        44
           1       0.97      0.92      0.95        76

    accuracy                           0.93       120
   macro avg       0.92      0.94      0.93       120
weighted avg       0.94      0.93      0.93       120
```

SVM

```
def svm_grid_search(X, y):
    #create a dictionary of all values we want to test
    param_grid = {'C': [0.1,1, 10, 100], 'gamma': [1,0.1,0.01,0.001, 0.4, 0.2, 0.8],'kernel': ['rbf', 'poly', 'sigmoid']}
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)

    svm = SVC()

    #use gridsearch to test all values
    svm_gscv = RandomizedSearchCV(estimator = svm,
                          param_distributions = param_grid,
                          scoring = 'accuracy',
                          cv = 4,
                          n_jobs = -1)
    #fit model to data
    svm_gscv.fit(X, y)

    return svm_gscv.best_params_
```

Over Sample SVM

```
from sklearn import svm
svm = SVC(gamma=0.8, C=10, kernel='rbf', probability=True)

svm.fit(X_train, y_train)

y_pred_svm = svm.predict(X_test)

print(svm.score(X_train, y_train))
print(svm.score(X_test, y_test))

print(metrics.classification_report(y_test, y_pred_svm))
```

```
1.0
0.9666666666666667
              precision    recall  f1-score   support

           0       0.98      0.93      0.95        44
           1       0.96      0.99      0.97        76

    accuracy                           0.97       120
   macro avg       0.97      0.96      0.96       120
weighted avg       0.97      0.97      0.97       120
```

XGBOOST

```
xgb = XGBClassifier(min_child_weight=1, max_depth=10, learning_rate=0.25, gamma=0.4, colsample_bytree=0.3)
xgb.fit(X_train,y_train)

y_pred_xgb = xgb.predict(X_test)

print(classification_report(y_test, y_pred_xgb))

print("Accuracy:",metrics.accuracy_score(y_test, y_pred_xgb))
print("Precision:",metrics.precision_score(y_test, y_pred_xgb))
print("Recall:",metrics.recall_score(y_test, y_pred_xgb))

print(xgb.score(X_train,y_train))
print(xgb.score(X_test,y_test))
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        44
           1       1.00      0.99      0.99        76

    accuracy                           0.99       120
   macro avg       0.99      0.99      0.99       120
weighted avg       0.99      0.99      0.99       120

Accuracy: 0.9916666666666667
Precision: 1.0
Recall: 0.986842105263158
1.0
0.9916666666666667
```

5. CONCLUSION

The proposed method for diagnosing CKD is supported by both data imputation and sample diagnosis. By employing KNN to impute missing data from the unsupervised data set, the integrated model was able to achieve satisfactory results. Given this, we postulate that using this strategy to make an accurate diagnosis of CKD will provide fruitful results. Furthermore, this approach may benefit from the clinical data from other illnesses utilized in actual medical diagnosis. However, due to limitations imposed by the criteria, we could only use 400 samples of the provided data in our model-building process. Because of this, it appears that the model's adaptability may be limited. In addition, there are only two categories of data samples in the dataset (ckd and notckd), hence the model cannot assess the severity of CKD. To improve generalization performance and disease severity identification, the model will be retrained in the future using a larger, more complex data set. We anticipate that when more and higher-quality data becomes available, our model will also progress.

6. REFERENCES

[1] "Diagnosis of patients with chronic kidney disease by using two fuzzy classifiers," Z. Chen et al., Chemometr. Intell. Lab., volume 153, pages 140-145, April 2016.

[2] Chronic kidney illness: a diagnostic challenge: A. Subasi, E. Alickovic, and J. Kevric, "Diagnosis of chronic kidney disease by using random forest," in Proc. Int. Conf. Medical and Biological Engineering, March 2017, pp. 589-594.

[3] Chronic renal disease prevalence in China: a cross-sectional survey, L. Zhang et al., Lancet, vol. 379, pp. 815–822, August 2012.

[4] Reference: Singh A. et al., "Incorporating temporal EHR data in predictive models for risk stratification of renal function deterioration," J. Biomed. Inform., vol. 53, pp. 220-228 (Feb. 2015).

[5] Citation: A. M. Cueto-Manzano et al., "Prevalence of chronic kidney disease in an adult population," Arch. Med. Res., vol. 45, no. 6, pp. 507-513, August 2014.

[6] H. Polat, H.D. Mehr, and A. Cetin, "Diagnosis of chronic kidney disease based on support vector machine by feature selection methods," J. Med. Syst., vol. 41, no. 4, Apr. 2017.

[7] According to "A new machine learning approach for predicting the response to anemia treatment in a large cohort of end stage renal disease patients undergoing dialysis," published in Comput. Biol. Med., vol. 61, pp. 56-61, Jun. 2015.

[8] According to Polat (2017), Danaei Mehr (2017), and Cetin (2017). Support vector machine-based chronic renal disease diagnosis via feature selection techniques. 41(4), 1-11 in the Journal of Medical Systems.

[9]Ibraheem Ibrahim and Abdulazeez Ali. Disease diagnosis using machine learning methods. 2(01):10-19 in Journal of Emerging Trends in Applied Sciences and Technologies.

[10] "Explainable prediction of chronic renal disease in the Colombian population using neural networks and case- based reasoning," IEEE Access, vol. 7, pp. 152900- 152910, G. R. Vasquez-Morales, S. M. Martinez-Monterrubio, P. MorenoGer, and J. A. Recio-Garcia, 2019.