

Model Development Phase Template

Date	15 MARCH 2024
Team ID	LTVIP2024TMID25011
Project Title	Early Prediction Of Chronic Kidney Disease Using Machine Learning
Maximum Marks	4 Marks

Initial Model Training Code, Model Validation and Evaluation Report

The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include classification reports, accuracy, and confusion matrices for multiple models, presented through respective screenshots.

Initial Model Training Code

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the model
model = RandomForestClassifier(n_estimators=20)
model.fit(X_train, y_train)

# Predict and calculate metrics
y_pred = model.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

# Print the confusion matrix and accuracy
print("Confusion Matrix:")
print(cm)
print(f"Accuracy is {round(accuracy_score(y_test, y_pred) * 100, 2)}%")

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Print the accuracy score
print(f"Accuracy is {round(accuracy_score(y_test, model.predict(X_test)) * 100, 2)}%")

# Print the confusion matrix
print("Confusion Matrix")
print(confusion_matrix(y_test, model.predict(X_test)))

# Print the classification report
print("Classification Report")
print(classification_report(y_test, model.predict(X_test)))
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.model_selection import train_test_split

# Assuming X_train, X_test, y_train, y_test are already defined
model = LogisticRegression()
model.fit(X_train, y_train)
pred = model.predict(X_test)

# Calculate the accuracy
accuracy = accuracy_score(y_test, pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Print the confusion matrix
print("Confusion Matrix:")
print(confusion_matrix(y_test, pred))

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, pred))
```

```
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Assuming X_train, X_test, y_train, y_test are already defined
model = GaussianNB()
model.fit(X_train, y_train)
pred = model.predict(X_test)

# Calculate the accuracy
accuracy = accuracy_score(y_test, pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Print the confusion matrix
print("Confusion Matrix:")
print(confusion_matrix(y_test, pred))

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, pred))
```

```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Assuming X_train, X_test, y_train, y_test are already defined
model = SVC()
model.fit(X_train, y_train)
pred = model.predict(X_test)

# Calculate the accuracy
accuracy = accuracy_score(y_test, pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Print the confusion matrix
print("Confusion Matrix:")
print(confusion_matrix(y_test, pred))

# Print the classification report
print("\nClassification Report:")
print(classification_report(y_test, pred))
```

```
import pickle
pickle.dump(model, open('kidney.pkl', 'wb'))
```

Model Validation and Evaluation Report:

Model	Classification Report	Accuracy	Confusion Matrix
-------	-----------------------	----------	------------------

Random Forest Classifier	<pre>from sklearn.model_selection import train_test_split from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import confusion_matrix, accuracy_score, classification_report # Split the data X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42) # Initialize and train the model model = RandomForestClassifier(n_estimators=20) model.fit(X_train, y_train) # Predict and calculate metrics y_pred = model.predict(X_test) cm = confusion_matrix(y_test, y_pred) # Print the confusion matrix and accuracy print("Confusion Matrix:") print(cm) print(f"Accuracy is (round(accuracy_score(y_test, y_pred) * 100, 2))%") # Print the classification report print("Classification Report:") print(classification_report(y_test, y_pred))</pre>	100.0%	<div>Confusion Matrix:</div> <div>[[23 0] [0 9]]</div> <div>Accuracy is 100.0%</div>																														
KNN	<pre>from sklearn.metrics import accuracy_score, confusion_matrix, classification_report # Print the accuracy score print(f"Accuracy is (round(accuracy_score(y_test, model.predict(X_test)) * 100, 2))%") # Print the confusion matrix print("Confusion Matrix") print(confusion_matrix(y_test, model.predict(X_test))) # Print the classification report print("Classification Report") print(classification_report(y_test, model.predict(X_test)))</pre>	100.0%	<div>Accuracy is 100.0%</div> <div>Confusion Matrix</div> <div>[[23 0] [0 9]]</div> <div>Classification Report</div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>1.00</td><td>1.00</td><td>1.00</td><td>23</td></tr><tr><td>1</td><td>1.00</td><td>1.00</td><td>1.00</td><td>9</td></tr><tr><td>accuracy</td><td></td><td></td><td>1.00</td><td>32</td></tr><tr><td>macro avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>32</td></tr><tr><td>weighted avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>32</td></tr></tbody></table>		precision	recall	f1-score	support	0	1.00	1.00	1.00	23	1	1.00	1.00	1.00	9	accuracy			1.00	32	macro avg	1.00	1.00	1.00	32	weighted avg	1.00	1.00	1.00	32
	precision	recall	f1-score	support																													
0	1.00	1.00	1.00	23																													
1	1.00	1.00	1.00	9																													
accuracy			1.00	32																													
macro avg	1.00	1.00	1.00	32																													
weighted avg	1.00	1.00	1.00	32																													
Logistic Regression	<pre>from sklearn.linear_model import LogisticRegression from sklearn.metrics import accuracy_score, confusion_matrix, classification_report from sklearn.model_selection import train_test_split # Assuming X_train, X_test, y_train, y_test are already defined model = LogisticRegression() model.fit(X_train, y_train) pred = model.predict(X_test) # Calculate the accuracy accuracy = accuracy_score(y_test, pred) print(f"Accuracy: {accuracy * 100:.2f}%") # Print the confusion matrix print("Confusion Matrix:") print(confusion_matrix(y_test, pred)) # Print the classification report print("\nClassification Report:") print(classification_report(y_test, pred))</pre>	96.88%	<div>Accuracy: 96.88%</div> <div>Confusion Matrix:</div> <div>[[23 0] [1 8]]</div> <div>Classification Report:</div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.96</td><td>1.00</td><td>0.98</td><td>23</td></tr><tr><td>1</td><td>1.00</td><td>0.89</td><td>0.94</td><td>9</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.97</td><td>32</td></tr><tr><td>macro avg</td><td>0.98</td><td>0.94</td><td>0.96</td><td>32</td></tr><tr><td>weighted avg</td><td>0.97</td><td>0.97</td><td>0.97</td><td>32</td></tr></tbody></table>		precision	recall	f1-score	support	0	0.96	1.00	0.98	23	1	1.00	0.89	0.94	9	accuracy			0.97	32	macro avg	0.98	0.94	0.96	32	weighted avg	0.97	0.97	0.97	32
	precision	recall	f1-score	support																													
0	0.96	1.00	0.98	23																													
1	1.00	0.89	0.94	9																													
accuracy			0.97	32																													
macro avg	0.98	0.94	0.96	32																													
weighted avg	0.97	0.97	0.97	32																													
Naïve Baye's	<pre>from sklearn.naive_bayes import GaussianNB from sklearn.metrics import accuracy_score, confusion_matrix, classification_report # Assuming X_train, X_test, y_train, y_test are already defined model = GaussianNB() model.fit(X_train, y_train) pred = model.predict(X_test) # Calculate the accuracy accuracy = accuracy_score(y_test, pred) print(f"Accuracy: {accuracy * 100:.2f}%") # Print the confusion matrix print("Confusion Matrix:") print(confusion_matrix(y_test, pred)) # Print the classification report print("\nClassification Report:") print(classification_report(y_test, pred))</pre>	100.0%	<div>Accuracy: 100.00%</div> <div>Confusion Matrix:</div> <div>[[23 0] [0 9]]</div> <div>Classification Report:</div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>1.00</td><td>1.00</td><td>1.00</td><td>23</td></tr><tr><td>1</td><td>1.00</td><td>1.00</td><td>1.00</td><td>9</td></tr><tr><td>accuracy</td><td></td><td></td><td>1.00</td><td>32</td></tr><tr><td>macro avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>32</td></tr><tr><td>weighted avg</td><td>1.00</td><td>1.00</td><td>1.00</td><td>32</td></tr></tbody></table>		precision	recall	f1-score	support	0	1.00	1.00	1.00	23	1	1.00	1.00	1.00	9	accuracy			1.00	32	macro avg	1.00	1.00	1.00	32	weighted avg	1.00	1.00	1.00	32
	precision	recall	f1-score	support																													
0	1.00	1.00	1.00	23																													
1	1.00	1.00	1.00	9																													
accuracy			1.00	32																													
macro avg	1.00	1.00	1.00	32																													
weighted avg	1.00	1.00	1.00	32																													
SVM	<pre>from sklearn.svm import SVC from sklearn.metrics import accuracy_score, confusion_matrix, classification_report # Assuming X_train, X_test, y_train, y_test are already defined model = SVC() model.fit(X_train, y_train) pred = model.predict(X_test) # Calculate the accuracy accuracy = accuracy_score(y_test, pred) print(f"Accuracy: {accuracy * 100:.2f}%") # Print the confusion matrix print("Confusion Matrix:") print(confusion_matrix(y_test, pred)) # Print the classification report print("\nClassification Report:") print(classification_report(y_test, pred))</pre>	81.25%	<div>Accuracy: 81.25%</div> <div>Confusion Matrix:</div> <div>[[23 0] [6 3]]</div> <div>Classification Report:</div> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0</td><td>0.79</td><td>1.00</td><td>0.88</td><td></td></tr><tr><td>1</td><td>1.00</td><td>0.33</td><td>0.50</td><td></td></tr><tr><td>accuracy</td><td></td><td></td><td>0.81</td><td></td></tr><tr><td>macro avg</td><td>0.90</td><td>0.67</td><td>0.69</td><td></td></tr><tr><td>weighted avg</td><td>0.85</td><td>0.81</td><td>0.78</td><td></td></tr></tbody></table>		precision	recall	f1-score	support	0	0.79	1.00	0.88		1	1.00	0.33	0.50		accuracy			0.81		macro avg	0.90	0.67	0.69		weighted avg	0.85	0.81	0.78	
	precision	recall	f1-score	support																													
0	0.79	1.00	0.88																														
1	1.00	0.33	0.50																														
accuracy			0.81																														
macro avg	0.90	0.67	0.69																														
weighted avg	0.85	0.81	0.78																														