Model Development Phase Template

Date	15 July 2024
Team ID	740145
Project Title	Doctors Annual Salary Prediction
Maximum Marks	10 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 a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

Initial Model Training Code (5 marks):

For the project titled "Doctor's Salary Prediction Using ML," the initial model training code is a crucial part of the development process. This code sets up the machine learning environment, prepares the data, and trains the model to predict doctors' salaries based on various features.

```
from pathlib import PureWindowsPath
import numpy as np import pandas as
pd import warnings
warnings.filterwarnings("ignore")
import matplotlib.pyplot as plt import
seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split,GridSearchCV df
= pd.read_excel(r'C:\Users\akhil\OneDrive\Desktop\soumyasri
project\Dataset\NewDoctorsPay.xlsx')


df  df.info()
df.isnull().sum() le
= LabelEncoder()
df['Specialty'] = le.fit_transform(df['Specialty'])
#df['Annual Income'] = le.fit_transform(df['Annual Income'])
```

```
df['Feel Fairly Compensated'] = le.fit_transform(df['Feel Fairly
```

```
Compensated'])
df['Overall Career Satisfaction'] = le.fit_transform(df['Overall Career Satisfaction'])
df['Satisfied Income'] = le.fit_transform(df['Satisfied Income']) df['Would Choose Medicine
Again'] = le.fit_transform(df['Would Choose Medicine Again']) df['Would Choose the Same
Specialty'] = le.fit_transform(df['Would Choose the Same Specialty'])
df['Survey Respondents by Specialty'] = le.fit_transform(df['Survey Respondents by Specialty'])
df.describe()
df.drop(columns=["% Female","Difference in Earnings between Physicians who Feels Fairly vs
Unfairly Paid"],axis=1,inplace=True) df.describe()
sns.countplot(df['Annual Income']) plt.show()
sns.boxplot(x='Annual Income',y='Overall Career
Satisfaction',data=df) plt.show() sns.pairplot(df) plt.show()
x = df.drop(['Annual Income'],axis=1) y
= df['Annual Income']
from sklearn.model_selection import train_test_split,GridSearchCV
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state = 42)
from sklearn.linear_model import LinearRegression reg = LinearRegression()
reg.fit(x_train,y_train)
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
y_train_pred = reg.predict(x_train)
y_test_pred = reg.predict(x_test)
y_train_pred[:5] y_test_pred[:5]
r2_score(y_train,y_train_pred) * 100
mean_squared_error(y_train,y_train_pre
d r2_score(y_test,y_test_pred)*100
mean_squared_error(y_test,y_test_pred)
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor(n_estimators=100,random_state=42) rf.fit(x_train,y_train)
y_train_pred=rf.predict(x_train)
y_test_pred=rf.predict(x_test)
r2_score(y_train,y_train_pred)
mean_squared_error(y_train,y_train_pred)
r2_score(y_test,y_test_pred)
mean_squared_error(y_test,y_test_pred) from
sklearn.tree import DecisionTreeRegressor
dtr=DecisionTreeRegressor(random_state=42)
```

```
dtr.fit(x_train,y_train)
y_train_pred=dtr.predict(x_train)
y_train_pred=dtr.predict(x_test)
y_train_pred[:5] y_train_pred[:5]
r2_score(y_train,y_train_pred)*100
r2_score(y_train,y_train_pred)*100
r2_score(y_train,y_train_pred)*100
mean_squared_error(y_test,y_test_pred)
import xgboost as xgb
xg_reg=xgb.XGBRegressor()
xg_reg.fit(x_train,y_train)
xg_reg.fit(x_train,y_train)
y_train_pred=xg_reg.predict(x_train)
y_test_pred=xg_reg.predict(x_test)
r2_score(y_train,y_train_pred)*100
mean_squared_error(y_train,y_train_pred)
r2_score(y_test,y_test_pred)*100
mean_squared_error(y_test,y_test_pred)
r2_score(y_train,y_train_pred)*100
reg.predict([[11,5,1,5,18,0,1]])
reg.predict([[23,9,6,9,1,12,4]])
reg.predict([[10,7,9,7,16,9,0]])
```

Model Validation and Evaluation Report (5 marks):

Model 1	**Linear Regression**: Simple linear model that assumes a linear relationship between the features and the target variable.	**Training MSE**: 1405231186.3519106 **Validation MSE**:3715045452.16928 **R-squared**: 27.269167796800964
Model 2	**Random Forest regression:**Random Forest Regression is a supervised machine learning algorithm that uses an ensemble of decision trees to predict continuous target variables	**Training MSE**: 458713655.5555556 **Validation MSE**: 3631440587.5 **R-squared**: 0.289059314547212
Model 3	**Decision tree regreesor :**A Decision Tree Regressor is a supervised machine learning algorithm used for predicting continuous target variables.	**Training MSE**: 0.0 **Validation MSE**:3565125000.0 **R-squared**: 100.0
Model 4	**XGBRegressor :** The `XGBRegressor` is a powerful machine learning algorithm from the XGBoost library, designed for regression tasks.	**Training MSE**:0.000132921006944444445 **Validation MSE**:3566414555.666992 **R-squared**:99.999999999

Model	Summary	Training and Validation Performance Metrics