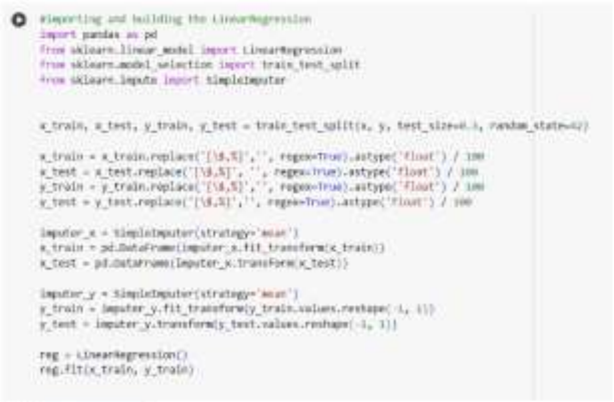
Model Development Phase Template

Date	15 July 2024
Team ID	740054
Project Title	Doctors Annual Salary Prediction
Maximum Marks	4 Marks

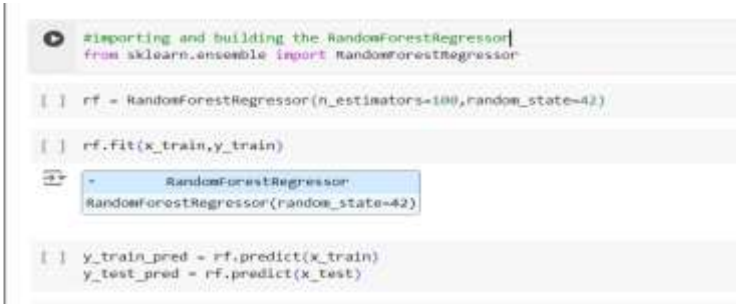
Initial Model Training Code, Model Validation and Evaluation Report

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:



```
#importing and building the LinearRegression
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.1, random_state=42)

x_train = x_train.replace('[\$,%]','', regex=True).astype('float') / 100
x_test = x_test.replace('[\$,%]', '', regex=True).astype('float') / 100
y_train = y_train.replace('[\$,%]','', regex=True).astype('float') / 100
y_test = y_test.replace('[\$,%]','', regex=True).astype('float') / 100

imputer_x = SimpleImputer(strategy='mean')
x_train = pd.DataFrame(imputer_x.fit_transform(x_train))
x_test = pd.DataFrame(imputer_x.transform(x_test))

imputer_y = SimpleImputer(strategy='mean')
y_train = imputer_y.fit_transform(y_train.values.reshape(-1, 1))
y_test = imputer_y.transform(y_test.values.reshape(-1, 1))

reg = LinearRegression()
reg.fit(x_train, y_train)
```



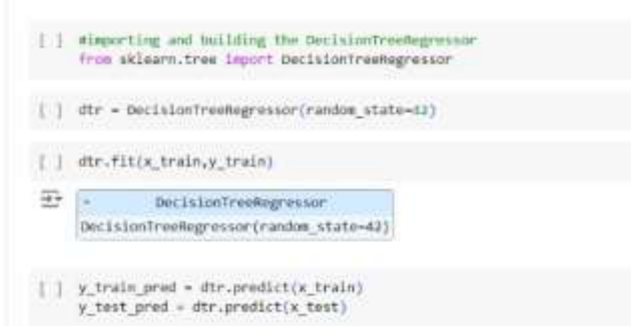
```
#importing and building the RandomForestRegressor
from sklearn.ensemble import RandomForestRegressor
```

```
rf = RandomForestRegressor(n_estimators=100,random_state=42)
```

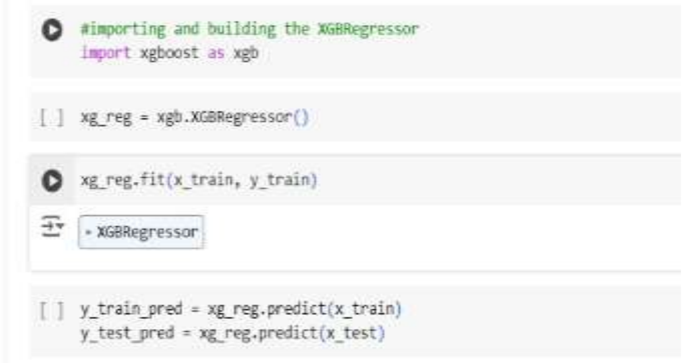
```
rf.fit(x_train,y_train)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

```
y_train_pred = rf.predict(x_train)
y_test_pred = rf.predict(x_test)
```

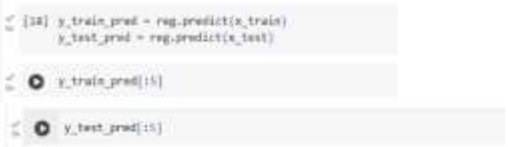
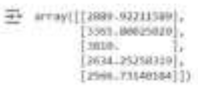


```
[ ] #importing and building the DecisionTreeRegressor
    from sklearn.tree import DecisionTreeRegressor

[ ] dtr = DecisionTreeRegressor(random_state=42)

[ ] dtr.fit(x_train,y_train)

        DecisionTreeRegressor
    DecisionTreeRegressor(random_state=42)

[ ] y_train_pred = dtr.predict(x_train)
    y_test_pred = dtr.predict(x_test)
```

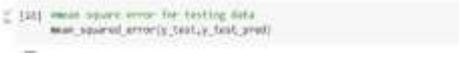


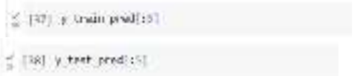
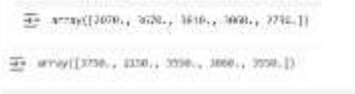
```
#importing and building the XGBRegressor
import xgboost as xgb

[ ] xg_reg = xgb.XGBRegressor()

xg_reg.fit(x_train, y_train)

    XGBRegressor

[ ] y_train_pred = xg_reg.predict(x_train)
    y_test_pred = xg_reg.predict(x_test)
```

Model Validation and Evaluation Report:

Model	Classification Report	Confusion Matrix
Linear Regression	 [18] y_train_pred = reg.predict(x_train) y_test_pred = reg.predict(x_test) y_train_pred[:5] y_test_pred[:5]	 array([[2889.92211589], [3365.80025020], [3810.], [2634.25258319], [2566.73140184]])

		 array([[2790.96681967], [3868.58519316], [3677.67142277], [3601.19452805], [3267.38001545]])

Random Forest Regressor	
	[38] #mean square error for testing data
	mean_squared_error(y_test,y_test_pred)
		
		371584.5852180281

Decision Tree Regressor	
	[37] y_train_pred[:5]
	[38] y_test_pred[:5]
		
		array([2070., 3620., 3610., 3060., 2710.])
		array([3750., 2350., 3550., 3060., 3550.])

XGB Regressor	 mean_squared_error(y_train,y_train_pred)	 7.669030507425533e-07