ASSIGNMENT 2- Applied Machine Learning (BUAN 6341.501)

Name: Darshil Sanghvi

Net ID: drs170530

I. Student Dataset – Binary Classification problem of predicting the Grade of the student based on the 30 independent variables of demographics data

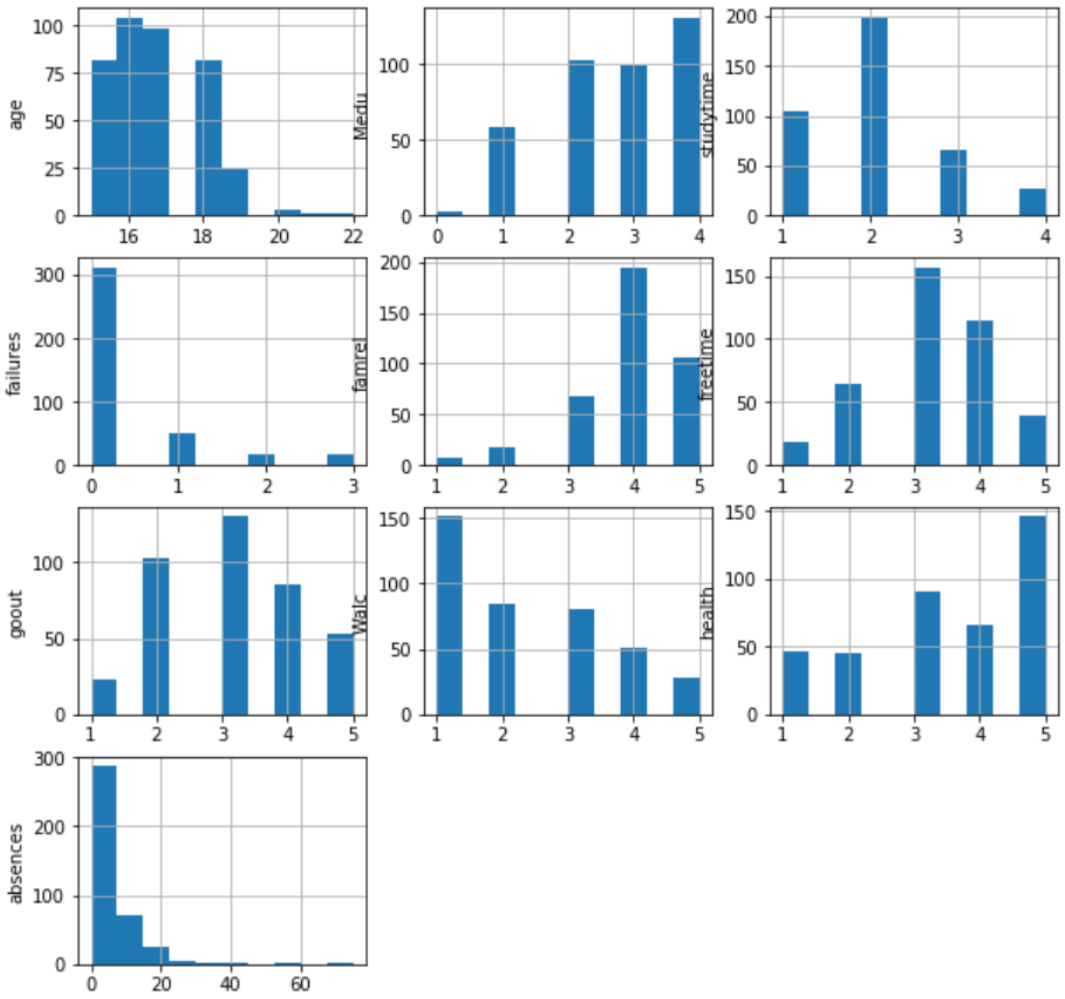
Step 1: **Variable Modifications** (dummy variables of categorical independent variables data)

Step 2: **Feature selection** using Recursive Feature Elimination. We are left with 10 independent variables after using this technique to find the significant variables out of 30

```
In [74]: rf_random.best_estimator_

Out[74]: RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=100,
                    max_features='auto', max_leaf_nodes=None,
                    min_impurity_decrease=0.0, min_impurity_split=None,
                    min_samples_leaf=4, min_samples_split=5,
                    min_weight_fraction_leaf=0.0, n_estimators=1000, n_jobs=1,
                    oob_score=False, random_state=None, verbose=0, warm_start=False)

In [75]: rfe_estimation = RFE(rf,n_features_to_select=10)
         rfe_estimation = rfe_estimation.fit(new_X, Y)

         C:\Users\VisualBI\Anaconda3\lib\site-packages\sklearn\utils\validation.py:578: DataConversionWarning: A column-vector y was pas
         sed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
           y = column_or_1d(y, warn=True)

In [76]: rfe_estimation.get_support
         features_bool = np.array(rfe_estimation.support_)
         features = np.array(new_X.columns)
         result = features[features_bool]
         print(result)

         ['age' 'Fedu' 'studytime' 'failures' 'famrel' 'freetime' 'goout' 'Walc'
          'health' 'absences']
```

These are those 10 features: `'age' ,'Fedu', 'studytime', 'failures', 'famrel', 'freetime', 'goout', 'Walc', 'health', 'absences'`

Step 3: **Data Wrangling**: There is a lot of skewness in the data as per the histograms and describing the mean and median of the data



Further, to tackle the outliers and skewness of 10 features, we perform various types of scaling (scaling, minmax scaler, normalization, standardization). We find normalization providing the best results:

```
The training score with normalization is : 0.15901409890395124
The testing score with normalization is 0.03729454204864402
```

Step 4**: Transforming Target variable into binary form**

################## binary classification of G3 i.e target variable #########

def f(row):

if row['G3'] > 10 :

val = 1

```
    else:
        val = 0
    return val
df['Grade'] = df.apply(f, axis=1)
df.head()
```

Step 5: Splitting the features into X and Y and further classifying the records into train and test

```
X = df_10[['age', 'Medu', 'studytime', 'failures', 'famrel', 'freetime', 'goout', 'Walc', 'health', 'absences']]
Y= df['Grade']
## Randomly grouping data into train and test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, random_state = 5)
```

Step 6: Applying three classification models with changing parameters and selecting the best one by looking at Accuracy_score and Error metrics (ROC curve)

Model	Accuracy_Score	Error (ROC_AUC Score)
1. SVM classification with linear kernel	49.495%	0.49
2. SVM classification with rbf kernel	52.525%	0.528
3. SVM classification with sigmoid kernel	49.495%	0.5
4. Decision Tree with 'Gini' metric without pruning	51.515%	0.51
5. Decision Treee with 'Gini' metric with pruning with tree depth = 8	52.525%	0.52
6. Boosting Ensemble methods with Decision tree without boosting	54.545%	0.54
7. Boosting Ensemble methods with Decision tree with pruning considering depth as 2	64.646%	0.64

Step 7: Interpretations:

SVM:

The performance and error metrics results of SVM remain same across all the 3 kernel changes

Decision Trees:

Impurity measure are quite consistent with each other... Indeed, the strategy used to prune the tree has a greater impact on the final tree than the choice of impurity measure. We choose Gini index as it focuses more on correct Classification than exploration (Entropy). Also, Gini index provides better performance

The performance (Accuracy_score) results improve after pruning

Ensemble Method using AdaBoosting:

We can see that the Model accuracy has increased considerably from 54% to 64% improving the performance after pruning/boosting.

Final Verdict: Boosting Ensemble methods with Decision tree with pruning performs the best in terms of Accuracy_Score of 64.64%.

II. Loan Dataset: Predict whether the person should be granted a loan or not based on his income, credit history and other demographics features.

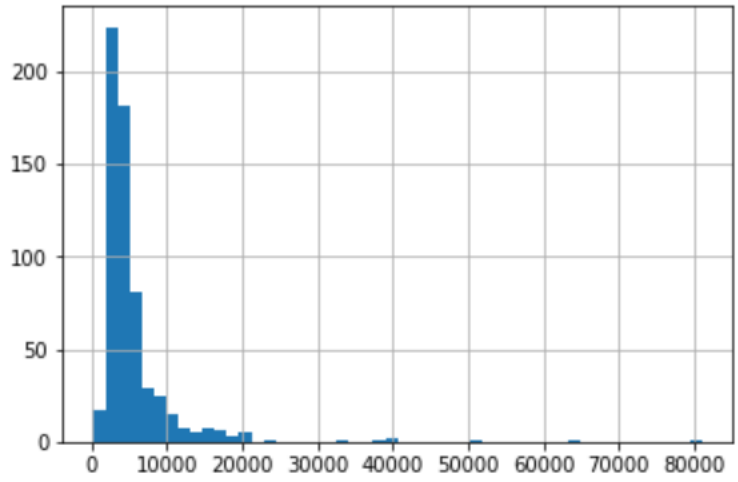
Target variable: Loan_Status

Predictors: Loan_ID, Gender, Married, Dependents, Education, Self-Employed, AppliantIncome, CoAppliantIncome, LoanAmount, LoanAmountTurn, Credit History, Property Area

Step 1: Data Exploratory Analysis

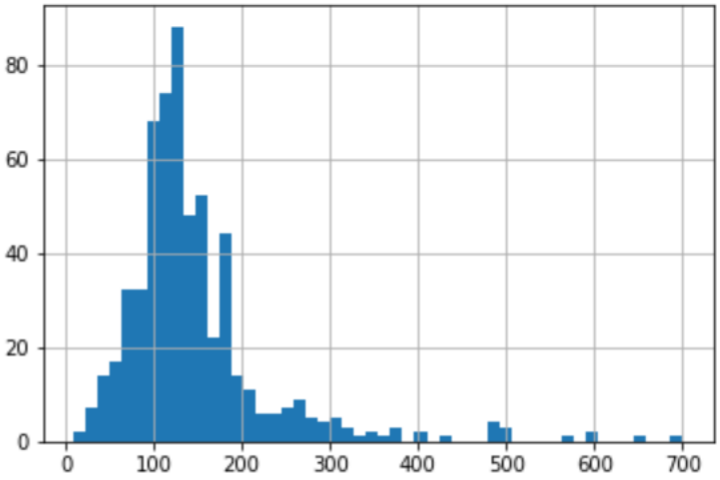
In [126]:
```
## Plotting histogram and box plots to study the distributions of num
df['ApplicantIncome'].hist(bins=50)
```

Out[126]: <matplotlib.axes._subplots.AxesSubplot at 0x1e3988e0828>



In [127]:
```
df.boxplot(column='ApplicantIncome')
```

```
: df['LoanAmount'].hist(bins=50)
: <matplotlib.axes._subplots.AxesSubplot at 0x1e398bc6dd8>
```




```
: df.boxplot(column='LoanAmount')
: <matplotlib.axes._subplots.AxesSubplot at 0x1e399d0d438>
```

Step 2: Data Wrangling:

1. **Imputing missing values** in the variables

Reidentifying the missing values:

df.apply(lambda x: sum(x.isnull()),axis=0)

imputing 'LoanAmount' with it's mean

df['LoanAmount'].fillna(df['LoanAmount'].mean(), inplace=True)

Since more than 80% values are “No”, it is safe to impute the missing values as “No”

df['Self_Employed'].fillna('No',inplace=True)

imputing others

df['Gender'].fillna(df['Gender'].mode()[0], inplace=True)

df['Married'].fillna(df['Married'].mode()[0], inplace=True)

df['Dependents'].fillna(df['Dependents'].mode()[0], inplace=True)

df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mode()[0], inplace=True)

```
df['Credit_History'].fillna(df['Credit_History'].mode()[0], inplace=True)
```

Data Transformation 2:

let's try a **log transformation** to nullify the skewed histograms of LoanAmount and ApplicantIncome

Step 3: **Data Modifications**:

CONVERTING all categorical variables to numerical before starting with predictions

```
var_mod =
['Gender','Married','Dependents','Education','Self_Employed','Property_Area','Loan_Status']

le = LabelEncoder()

for i in var_mod:

   df[i] = le.fit_transform(df[i])

df.dtypes
```

Step 4**: Checking for cross-validation score error as the dataset is small.**

Performing k-fold cross-validation with 5 folds as it will help generalize the model better, help the model become skilled and test different types of models

#Generic function for making a cross validation model and accessing performance:

def cross_validation(model, data, X_train, Y_train):

{ …….

}

Step 5: ## Splitting data into X and Y

Grouping data into Train and Test

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y,test_size = 0.20, random_state =0)
```

Step 6: Applying three classification models with changing parameters and selecting the best one by looking at Accuracy_score and Error metrics (ROC curve)

Model	Accuracy_Score	Error (ROC_AUC Score)	Cross-Validation Score
1. SVM classification with linear kernel	82.927%	0.70	80.946%
2. SVM classification with rbf kernel	80.488%	0.67	77.528%
3. SVM classification with sigmoid kernel	73.171%	0.5	68.729%
4. Decision Tree with 'Gini' metric without pruning	64.228%	0.62	69.058%
5. Decision Treee with 'Gini' metric with pruning with tree depth = 8	75.610%	0.67	77.361%
6. Boosting Ensemble methods with Decision tree without boosting	66.667%	0.65	69.544%
7. Boosting Ensemble methods with Decision tree with pruning considering depth as 2	72.358%	0.647	74.926%

Step 7: Interpretations:

SVM: SVM with rbf kernel reduced accuracy and cross-validation score compared to linear kernel. Although, SVM with sigmoid kernel reduced accuracy and cross-validation score considerably, the error metric of ROC_AUC Score is 0.5 which denotes that it is a balanced dataset.

Decision Trees: Impurity measure are quite consistent with each other. Indeed, the strategy used to prune the tree has a greater impact on the final tree than the choice of impurity measure. We choose Gini index as it focuses more on correct Classification than exploration (Entropy). Also, Gini index provides better performance. The performance (Accuracy_score) results improve after pruning.

Ensemble Method using AdaBoosting: As expected the Test_Accuracy of the model inreases to 72.35% improving the performance due to pruning.

FINAL VERDICT: SVM has higher accuracy_score compared to Decision Tree and Ensemble Boosting method, but it carries the burden of high error metric. Except for the sigmoid kernel SVM model which has highest accuracy and lowest error score.