Income Prediction AI Project Report

1. Data Preprocessing:

➢ Import Libraries :

- Pandas library is used to handle data in tabular format.
- Numpy library provides numerical operations and array manipulations.
- Matplotlib library is used for data visualization.
- Seaborn library is used for enhanced data visualization.
- MinMaxScaler from sklearn.preprocessing is used for feature scaling.
- chi2, SelectKBest, mutual_info_classif, f_classif from sklearn.feature_selection are used for feature selection.
- kendalltau from scipy.stats is used to compute Kendall's rank correlation coefficient.
- train_test_split from sklearn.model_selection is used to split the dataset into training and testing sets.

➢ Data Loading and Concatenation:

- The train and test data are loaded using pd.read_csv() function.
- pd.concat() function is used to concatenate the train and test data into a single dataframe.

➢ Handling Missing Values:

- Missing values are checked using df.isnull().sum().
- For each column, the number of missing values represented as " ?" is calculated using df[c].isin([" ?"]).sum().
- Columns with missing values are identified, and the missing values are replaced with the mode of each column using df[col].replace(" ?", mode).

➢ Handling Categorical Variables:

- Categorical variables are converted to numerical codes using pd.Categorical(df[col]).codes.

Feature Scaling:

- MinMaxScaler from sklearn.preprocessing is used to scale the features. The scaler is fit on the training data using scaler.fit_transform(), and the scaled features are stored in df_scaled.

Outlier Detection and Removal:

- Outliers are detected using the interquartile range (IQR) method.
- The outliers are replaced with the median value using df_out = df[~(df < (Q1 - 1.5 * IQR) | (df > (Q3 + 1.5 * IQR))).any(axis=1)].
- The distribution of the data before and after removing outliers is visualized using box plots.

2. Feature Selection:

SelectKBest with Chi-Square:

- SelectKBest(score_func=chi2, k=10) is used to select the top 10 features based on the chi-square score.
- The fit() function is called on the SelectKBest object with X_train and y_train as input.
- The scores for each feature are stored in dfscores, and the corresponding feature names are stored in dfcolumns.
- The feature scores are concatenated into a single dataframe feature_scores.
- The top 15 features with the highest scores are printed using feature_scores.nlargest(15, 'Score').

Mutual Information:

- mutual_info_classif is used to calculate the mutual information between each feature and the target variable.
- The mutual information scores are stored in mutual_info.
- The scores are sorted in descending order, and the top 15 features with the highest scores are printed using mutual_info_scores.head(15).

Kendall's Rank Correlation Coefficient:

- The kendalltau function is used to compute Kendall's rank correlation coefficient between each feature and the target variable.
- The correlation coefficients are stored in the corr list.
- The correlation values are stored in corr_df dataframe along with the feature names.
- The features are sorted based on their correlation values using corr_df.sort_values().

- The top k (15 in this case) features with the highest correlation values are selected and stored in the selected_features list.
- The X_train data is subsetted using the selected features and stored in X_selected dataframe.

3. Classifiaction:

➢ Decision Tree Classifier:

- A decision tree classifier is trained using tree.DecisionTreeClassifier() with entropy as the criterion and max_depth of 3.
- The decision tree is visualized using plot_tree() function.
- The accuracy score and classification report are printed using decisionTree.score() and classification_report().

➢ Support Vector Machine (SVM):

- SVM models with different C values (1 and 33) are trained using SVC() from sklearn.svm.
- The accuracy and classification report are printed for each model using classification_report().

➢ Logistic Regression:

- Logistic regression model is trained using LogisticRegression() with a tolerance of 0.0001 and C value of 100.
- The predictions are made on the test set using LR.predict().
- The classification report is printed using classification_report().

➢ Accuracy Comparison:

- The decision tree model accuracy is: 84%
- The SVM model accuracy is: 85%
- The logistic regression model accuracy is: 82%

The highest and most efficient accuracy from the SVM model