Supervised Learning - Home Project

Regression Problem:

The **'Covid_19_GDP.xlsx'** dataset was collected from kaggle and originates from Mendeley Data: The Impact of Covid-19 Pandemic on the Global Economy: Emphasis on Poverty Alleviation and Economic Growth. The data consists of records on the impact of covid-19 on the global economy including 210 countries.

Your main goal is to predict the gdp_per_capita (label) of a country is affected due to COVID-19.

Dataset Columns explanation:

- **iso_code** → country code
- **location** → name of the country
- **date**
- **total_cases** → number of COVID19 cases
- **total_deaths**
- **stringency_index** → The Stringency Index provides a computable parameter to evaluate the effectiveness of the nationwide lockdown. It is used by the Oxford COVID-19 Government Response Tracker with a database of 17 indicators of government response such as school and workplace closings, public events, public transport, stay-at-home policies. The Stringency Index is a number from 0 to 100 that reflects these indicators. A higher index score indicates a higher level of stringency.
- **population**
- **gdp_per_capita** → A country's GDP or gross domestic product is calculated by taking into account the monetary worth of a nation's goods and services after a certain period of time, usually one year. It's a measure of economic activity.
- **hdi** → The HDI was created to emphasize that people and their capabilities should be the ultimate criteria for assessing the development of a country, not economic growth alone. The Human Development Index (HDI) is a summary measure of average achievement in key dimensions of human development: a long and healthy

life, being knowledgeable and having a decent standard of living. The HDI is the geometric mean of normalized indices for each of the three dimensions.

Data Preparation:

1. Remove unknown columns and meaningless columns from the dataset, for every column you removed provide an explanation.
2. Check for missing values in the dataset and decide how to handle them.
   Provide an explanation of the method you choose to handle the rows with missing values.
3. Check for duplicate rows in the dataset and remove them.
4. Convert categorical features to numerical form using the get_dummies() method if needed.

Data exploration:

1. Check the correlation between each feature to the label column and each feature to the other feature.
2. Generate a heatmap that visualizes those correlations.
3. Provide meaningful insight from the correlation check, which features have strong correlation to the label and which features have strong correlation with each other. Try to provide an explanation why you think those correlations are strong.

Linear Regression model training:

For each model perform the following:

1. Perform feature scaling of type Standardization.
2. Perform train / test split with test size → 0.3 and seed → 42
3. For finding optimal parameter values you can use any technique learned (Cross Validation / Grid Search / Elbow method, etc…).
   Make sure you show your calculations in your notebook.

Multi model training:

1. Perform **Vanilla Linear Regression** with K-fold cross validation
2. Perform **RidgeCV Regression** with optimal lambda parameter, show your calculation to find the optimal parameter,
3. Perform **LassoCV Régression** with optimal lambda parameter, show your calculation to find the optimal parameter,
4. Perform **Polynomial Regression** with optimal polynomial degree, show your calculation to find the optimal degree.

Multi model evaluation and deployment:

1. Print the optimal parameters values of each model (in case it has optimal values).
2. Print the beta coefficient the model found for each feature.
3. Evaluate each optimal model using **MAE**, **MSA** and **RMSE**
4. Generate a single plot that shows all models' accuracy values.
5. Choose the model that provided the most accurate prediction
6. Train the optimal model on the entire dataset.
7. Export your final model into a joblib file.
   Make sure you also export other relevant preprocessing instances such as the polynomial converter and the standard scaler.
8. Import your final model and the preprocessing instances from the joblib files and load them back to your working area.

Classification Problem:

Customer churn refers to the phenomenon where customers discontinue their relationship or subscription with a company or service provider. It represents the rate at which customers stop using a company's products or services within a specific period. Churn is an important metric for businesses as it directly impacts revenue, growth, and customer retention.

In the context of the Churn dataset, the churn label indicates whether a customer has churned or not. A churned customer is one who has decided to discontinue their subscription or usage of the company's services. On the other hand, a non-churned customer is one who continues to remain engaged and retains their relationship with the company.

Understanding customer churn is crucial for businesses to identify patterns, factors, and indicators that contribute to customer attrition. By analyzing churn behavior and its associated features, companies can develop strategies to retain existing customers, improve customer satisfaction, and reduce customer turnover. Predictive modeling techniques can also be applied to forecast and proactively address potential churn, enabling companies to take proactive measures to retain at-risk customers.

The **'customer_churn_dataset.csv'** file is a dataset that provides data about different customers and if they churned or not.

Your main goal is to predict if a future customer will likely to churn or not (label) based on the dataset features.

Dataset Columns explanation:

- **Age** → Age of the customer
- **Gender** → Gender of the customer (Male/Female)
- **Tenure** → Number of months the customer has stayed with the company
- **Usage Frequency** → How frequently the customer uses the service
- **Support Calls** → Number of calls the customer has made to customer support
- **Payment Delay** → Number of times the customer has delayed payment
- **Subscription Type** → Type of subscription the customer has

- **Contract Length** → Length of the customer's contract in months
- **Total Spend** → Total amount of money spent by the customer
- **Last Interaction** → Time since the last interaction with the customer
- **Churn** → Whether the customer churned (Yes/No)

Data Preparation:

1. Remove meaningless columns from the dataset, for every column you removed provide an explanation.
2. Check for missing values in the dataset and decide how to handle them.
   Provide an explanation of the method you choose to handle the rows with missing values.
3. Check for duplicate rows in the dataset and remove them.
4. Convert categorical features to numerical form using the get_dummies() method if needed.

Data exploration:

1. Check the correlation between each feature to the label column and each feature to the other feature.
2. Generate a pairplot that visualizes those correlations.
3. Provide meaningful insight from the correlation check, which features have strong correlation to the label and which features have strong correlation with each other.
   Try to provide an explanation why you think those correlations are strong.

Classification model training:

For each model perform the following:

1. Perform feature scaling of type Standardization (if needed).
2. Perform train / test split with test size → 0.3 and seed → 42
3. For finding optimal parameter values you can use any technique learned (Cross Validation / Grid Search / Elbow method, etc…).
   Make sure you show your calculations in your notebook.

Multi model training:

1. Perform **Logistic Regression** with optimal parameters.
2. Perform **KNN** with optimal K value.
3. Perform **SVM** with optimal parameters.
4. Perform **Random Forest** with optimal values for each decision tree and with optimal values for the Random Forest model itself (number of trees, bootstrapping etc…)

Multi model evaluation and deployment:

1. Print the optimal parameters values of each model (in case it has optimal values).
2. Print the actual predictions for the test set and the percentage of belonging to each class (in case the model provide this information)
3. Evaluate each optimal model using **Accuracy**, **Recall** and **F1-Score**.
4. Print for each model the corresponding confusion matrix as a heatmap plot (as learned in the course).
5. Generate a dataframe that contains each model and its values for the **Accuracy**, **Recall** and **F1-Score** metrics.
6. Choose the model that provided the most accurate prediction.
7. Train the optimal model on the entire dataset.
8. Export your final model into a joblib file.
   Make sure you also export other relevant preprocessing instances such as the polynomial converter and the standard scaler.
9. Import your final model and the preprocessing instances from the joblib files and load them back to your working area.

GOOD LUCK!

ECOM SCHOOL
המכללה למקצועות הדיגיטל וההייטק