Practical 2

Problem Statement:

To Predict the price of the Uber ride from a given pickup point to the agreed drop-off location.
Perform following tasks:

1. Pre-process the dataset.
2. Identify outliers.
3. Check the correlation.
4. Implement linear regression and ridge, Lasso regression models.
5. Evaluate the models and compare their respective scores like R2, RMSE,

Dataset: Uber fares Dataset.

Dataset link: https://www.kaggle.com/datasets/yasserh/uber-fares-dataset

Objective:

1. Understand working of linear regression and ridge, Lasso regression models.
2. Evaluate the models and compare their respective scores like R2, RMSE, etc.

Outcome:

Linear regression, ridge and Lasso regression models for prediction of test data should be understood and accuracy is evaluated.

Theory:

Linear regression is a type of statistical analysis used to predict the relationship between two variables. It assumes a linear relationship between the independent variable and the dependent variable, and aims to find the best-fitting line that describes the relationship. The line is determined by minimizing the sum of the squared differences between the predicted values and the actual values.

Linear regression is commonly used in many fields, including economics, finance, and social sciences, to analyse and predict trends in data. It can also be extended to multiple linear regression, where there are multiple independent variables, and logistic regression, which is used for binary classification problems.

Linear regression (in scikit-learn) is the most basic form, where the model is not penalized for its choice of weights, at all. That means, during the training stage, if the model feels like one particular feature is particularly important, the model may place a large weight to the feature.

This sometimes leads to overfitting in small datasets. Hence, following methods are invented.

Lasso is a modification of linear regression, where the model is penalized for the sum of absolute values of the weights. Thus, the absolute values of weight will be (in general) reduced, and many will tend to be zeros. During training, the objective function become:

$$\frac{1}{2m}\sum_{i=1}^{m}(y - Xw)^2 + alpha\sum_{j=1}^{p}|w_j|$$

As you see, Lasso introduced a new hyperparameter, alpha, the coefficient to penalize weights.

Ridge takes a step further and penalizes the model for the sum of squared value of the weights. Thus, the weights not only tend to have smaller absolute values, but also really tend to penalize the extremes of the weights, resulting in a group of weights that are more evenly distributed. The objective function becomes:

$$\sum_{i=1}^{n}(y - Xw)^2 + alpha\sum_{j=1}^{p}w_j^2$$

Requirements:

1) Get the Uber fare dataset

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

2) Importing Libraries

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

```
# importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

```

3) Importing the Datasets

Now we need to import the datasets which we have collected for our machine learning project.

```
df = pd.read_csv('uber-fares-dataset')
print(df.head())
```

4) Data Preprocessing and Cleaning

Here we handle missing values, if necessary, remove outliers, and converting data types as needed.

a) Checking where is null value

```
df.isnull().sum()
```

b) Replace null value by mean and median value

```
df['dropoff_latitude'].fillna(value=df['dropoff_latitude'].mean(),inplace = True)
df['dropoff_longitude'].fillna(value=df['dropoff_longitude'].median(),inplace = True)
```

c) Display the data types of each column in the DataFrame

```
df.dtypes
```

d) Change the Incorrect data type.

```
df.pickup_datetime = pd.to_datetime(df.pickup_datetime)
df.dtypes
```

d) drop the column 'key' ,'Unnamed: 0' and 'pickup_daetime'

```
df = df.drop(['key' ,'Unnamed: 0', 'pickup_datetime'],axis=1)
```

e) Outliers are data points that significantly deviate from the overall pattern or distribution of the dataset. They can be exceptionally high or low values that may distort statistical analysis or model performance, and they are important to identify and handle appropriately to avoid biased results or skewed interpretations of the data.

```
# Boxplot to check the outliers
# Resolving the outliers
#IQR to fill in the blanks
```

f) Correlation

quantifies the relationship between two variables. It indicates the degree to which the variables tend to change together, ranging from -1 (strong negative correlation) to 1 (strong positive correlation), with 0 indicating no linear relationship.

```
corr = df.corr()
```

5) Splitting out data to train and test using sklearn

```
X_train, X_test, y_train, y_test = train_test_split(x_std, y_std,
test_size=0.2, random_state=0)
```

6) Applying simple regression , lasso and Ridge model and Evaluate models

```
l_reg = LinearRegression()
l_reg.fit(X_train, y_train)
evaluate model performance on test data
y_pred = l_reg.predict(X_test)
```

```
lasso = Lasso(alpha=0.1)
lasso.fit(X_train, y_train)
```

```
y_pred = lasso.predict(X_test)
```

```
    model = RidgeRegression( iterations = 1000,
                             learning_rate = 0.01, l2_penality = 1 )
    model.fit( X_train, Y_train )

    # Prediction on test set
    Y_pred = model.predict( X_test )
```

7) Model Evaluation

There are three primary metrics used to evaluate linear models. These are: Mean absolute error (MAE), Mean squared error (MSE), or Root mean squared error (RMSE).

```
# Import metrics library
from sklearn import metrics

# Print result of MAE
print(metrics.mean_absolute_error(y_test, y_pred))

# Print result of MSE
print(metrics.mean_squared_error(y_true, y_pred))

# Print result of RMSE
print(np.sqrt(metrics.mean_squared_error(y_true, y_pred)))
```

Algorithm:

Linear regression

1. Data Pre-processing
2. Fitting the Simple Linear Regression to the Training Set
3. Prediction of test set result
4. visualizing the Training set results
5. visualizing the Test set results

Lasso regression

1. Data Pre-processing
2. Fitting the lasso regression to the Training Set
3. Prediction of test set result
4. visualizing the Training set results
5. visualizing the Test set results

Ridge regression

1. Data Pre-processing
2. Fitting the ridge regression to the Training Set
3. Prediction of test set result
4. visualizing the Training set results
5. visualizing the Test set results

Conclusion:

Linear regression the most basic form, where the model is not penalized for its choice of weights, at all. Ridge and Lasso's regression are a powerful technique for regularizing linear regression models and preventing overfitting. They both add a penalty term to the cost function, but with different approaches.

Oral Questions:

1. What is gradient descent?
2. What is cost function?
3. How Linear regression works?
4. What is difference between linear, lasso and ridge regression?