Machine Learning Engineer Assessment(RDCX)

As part of the machine learning assessment, I have built a **Machine Learning Model** which will predict **House Price** based on various features and deployed the model to **Heroku.** Heroku is a platform as a service (PaaS) that enables developers to build, run, and operate applications entirely in the cloud. So, whenever users will key in the attributes of different features, the App will give you the price estimations.

Fig: 1 User interface

Datasets:
I have taken the Boston House Pricing dataset which we can import from Skelarn. The dataset has 506 number of instances and 14 attribute information. This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University.

Libraries Used: Pandas, Numpy, Matplotlib, Pickle, Seaborn, json, Flask.
Tools Used: Google Colab, Visual Studio Code, Postman, Docker, Heroku
Language Used: Python and HTML

Methodology:

1. **Data Pre-processing:**

   i) As a crucial part of data preprocessing, I have tried to find the null values using isnull() function, there are no null values in the dataset, I have also seen the outliers but they look proper for modeling. Using describe () and info () function, I have checked the description of different columns for the data and data type.

2. **Exploratory Data Analysis (EDA):**

   i) Firstly, using Seaborn.pairplot, I have drawn a diagram to understand correlation between various column of the dataset, it is important because I need to see which columns are necessary for my target value, here in this dataset, the 'Price' is the target value. I have found some of the attributes are positively and negatively correlated with my target value.

   ii) To understand more, I have drawn some scatter plots with our target data to understand the correlation methods.

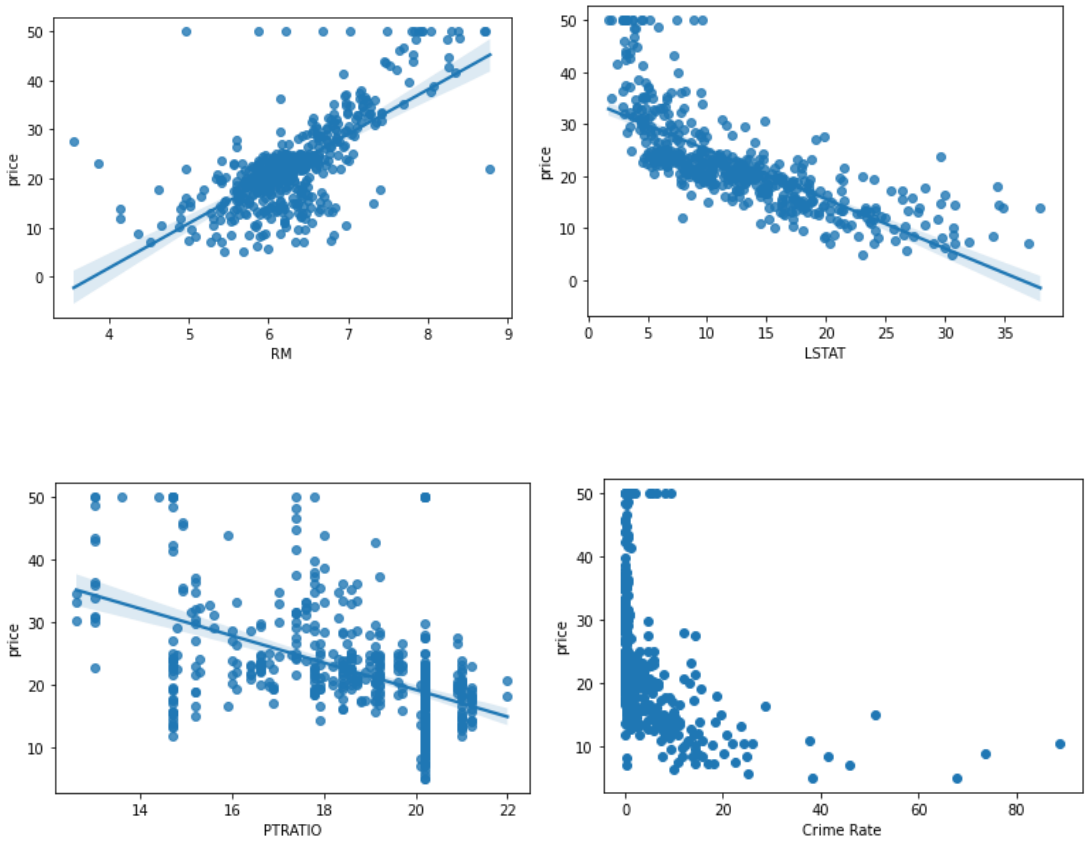

Fig 2: Understanding the Correlation

3. Creating our Models

i) Before creating a model, I have separated the dataset into train and test splits using sklearn.model_selection.

ii) Next, I have standardized my data using StandScalar from Skelarn, Data standardization helps improve the quality of your data by transforming and standardizing it. Standardization is an important technique that is mostly performed as a pre-processing step before many machine learning models, to standardize the range of features of an input data set.

iii) After completing all the techniques of data preprocessing and preparing the data for modeling, I have proceeded to the training phase. As it's Linear regression data, I have used the Linear Regression algorithm to train my model. I can use several other algorithms and choose the best one, but because this assessment is a time bound assessment and Linear Regression usually performs well with this sort of data, that's why I choose this algorithm. Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.
See below the scatterplot which is predicted by my trained model.

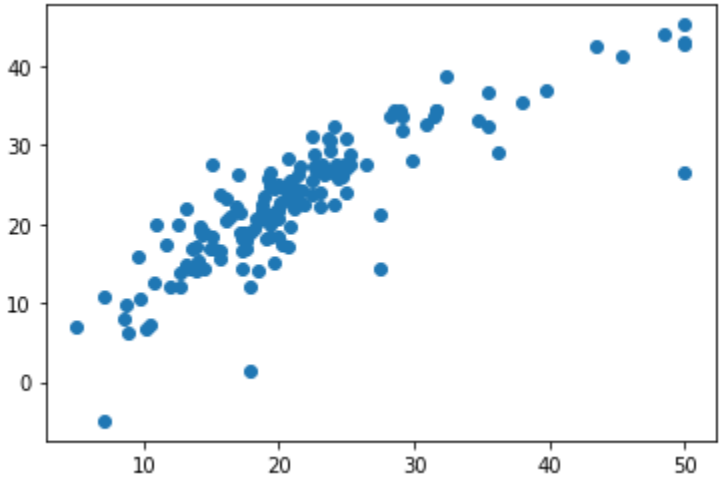

Fig 3: scatter plot for the prediction

4. **Accuracy Measurement:**

i) As part of the accuracy measurement, I have used 3 parameters which are mean absolute error, mean squared error and R squared and adjusted R square.

Mean Absolute error	3.744
Mean squared error	24.134
R squared	4.91

Finally, after getting the results from the model, I have pickled my model for further deployment.

Deployment:

For the deployment, I have created a Flask Rest API, with my trained model and then I have used Postman to test my API and finally I have created one Procfile and one Dockerfile to dockerize my model so that the infrastructure of the model remains same for every operating system, After finishing all of these steps, I have deployed my model to Heroku Cloud Platform, so the user can access the App.

1. **Flask API:**
   Flask is a popular micro framework for building web applications. Since it is a micro-framework, it is very easy to use and lacks most of the advanced functionality which is found in a full-fledged framework. Therefore, building a REST API in Flask is very simple.

2. **Postman:**
   Postman is an application used for API testing. It is an HTTP client that tests HTTP requests, utilizing a graphical user interface, through which we obtain different types of responses that need to be subsequently validated.

3. **Docker:**
   Docker is an open-source platform that enables developers to build, deploy, run, update and manage containers—standardized, executable components that combine application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.
4. **Heroku**
   Heroku is a container-based cloud Platform as a Service (PaaS). Developers use Heroku to deploy, manage, and scale modern apps. Our platform is elegant, flexible, and easy to use, offering developers the simplest path to getting their apps to market.

CI/CD Pipeline Using GitHub Actions:

As the whole project will be uploaded to GitHub that's why I have created a CI/CD pipeline (continuous integration/continuous delivery). This pipeline is very helpful because it will Ensure superior code quality, Deliver faster with an accelerated release rate, Continuous feedback and Optimum transparency and accountability.

26 lines (24 sloc) | 790 Bytes

```yaml
# Your workflow name.
name: Deploy to heroku.

# Run workflow on every push to main branch.
on:
  push:
    branches: [main]

# Your workflows jobs.
jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      # Check-out your repository.
      - name: Checkout
        uses: actions/checkout@v2
      - name: Build, Push and Release a Docker container to Heroku. # Your custom step name
        uses: gonuit/heroku-docker-deploy@v1.3.3 # GitHub action name (leave it as it is).
        with:
          email: ${{ secrets.HEROKU_EMAIL }}
          heroku_api_key: ${{ secrets.HEROKU_API_KEY }}
          heroku_app_name: ${{ secrets.HEROKU_APP_NAME }}
          dockerfile_directory: ./
          dockerfile_name: Dockerfile
          docker_options: "--no-cache"
          process_type: web
```

Fig 4: CI/CD pipeline

GitHub link for the project:
https://github.com/zianafique/house_pricing_deployment
House Price Prediction App:
https://price-house-prediction.herokuapp.com/
You can also access this app, by going to the GitHub repo and click on the Environments(price-house-predictions) and then see the deployments from there.