Internship Weekly Report – Week 4

◈ Title Page

Name: Sandeep Ravaji Patel
Domain: Data Science
Week Number: Week 4

◈ Task Description

Objective:
To understand the fundamentals of machine learning and apply them by building a simple linear regression model using Scikit-Learn, including data preprocessing, model training, and evaluation.

Tasks Completed:

Machine Learning Basics:

- Learned the difference between supervised and unsupervised learning.
- Focused on linear regression as a foundational algorithm in supervised learning.
- Understood concepts such as features, labels, training data, testing data, and overfitting.

Model Building:

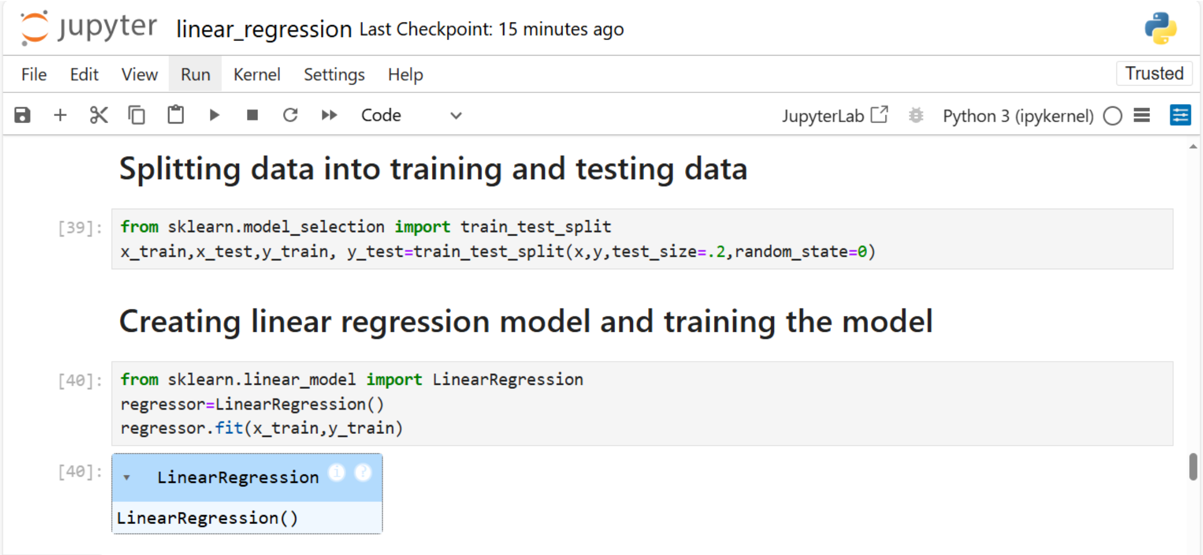
- Implemented **Linear Regression** using the 1000_Companies.csv dataset.
- Preprocessed data (handled categorical features using Label encoding).
- Split the dataset into training and testing sets using train_test_split.
- Trained the linear regression model using Scikit-Learn's LinearRegression() class.
- Predicted results and evaluated model performance using metrics like **R² Score** and **Mean Squared Error (MSE)**.

Tools Used:

- Scikit-Learn
- Pandas
- NumPy
- Jupyter Notebook

❖ Code Snippets / Design Screenshots

Example 1: Data Splitting and Model Training


jupyter
linear_regression Last Checkpoint: 15 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

Splitting data into training and testing data

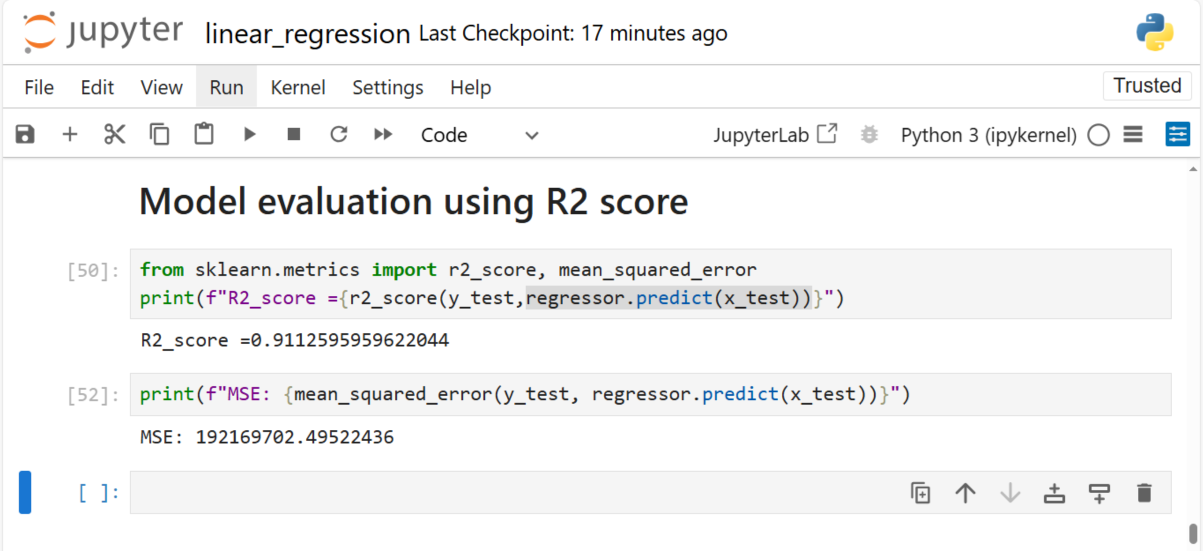
```
[39]: from sklearn.model_selection import train_test_split
      x_train,x_test,y_train, y_test=train_test_split(x,y,test_size=.2,random_state=0)
```

Creating linear regression model and training the model

```
[40]: from sklearn.linear_model import LinearRegression
      regressor=LinearRegression()
      regressor.fit(x_train,y_train)

[40]: ▾ LinearRegression
      LinearRegression()
```

Example 2: Model Evaluation


jupyter
linear_regression Last Checkpoint: 17 minutes ago
File Edit View Run Kernel Settings Help
Trusted
Code
JupyterLab
Python 3 (ipykernel)

Model evaluation using R2 score

```
[50]: from sklearn.metrics import r2_score, mean_squared_error
      print(f"R2_score ={r2_score(y_test,regressor.predict(x_test))}")

      R2_score =0.9112595959622044

[52]: print(f"MSE: {mean_squared_error(y_test, regressor.predict(x_test))}")

      MSE: 192169702.49522436

[ ]:
```

❖ Challenges Faced

1. Categorical Feature Handling:

- Dataset included non-numeric features that couldn't be directly used in model training.
- **Resolution:** Used LabelEncoder() from sklearn.preprocessing for Label encoding.

2. Model Accuracy Variability:

- Initial model showed low R^2 score.
- **Resolution:** Reviewed feature selection and normalized relevant features.

3. Data Splitting Concerns:

- Model was sensitive to random splits.
- **Resolution:** Used random_state in train_test_split for reproducibility.

❖ Learning Outcome

- Understood the end-to-end workflow of building a regression model.
- Gained confidence using Scikit-Learn for model training and evaluation.
- Learned how to handle real-world data challenges like feature encoding and data splitting.
- Interpreted regression metrics like R^2 score and MSE for model assessment.

❖ Next Steps

For **Week 5**, the focus will be on:

- Working on classification and regression models like Decision Tree and Logistic Regression.
- Evaluating models using accuracy and confusion matrix.
- Expanding understanding of model evaluation for classification problems.

❖ Resources

- **ML Basics:** Machine Learning with Python
- **Scikit-Learn Documentation:** https://scikit-learn.org/stable/
- **Dataset Used:** 1000_Companies.csv
