To,
IITD-AIA Foundation of Smart Manufacturing

Subject: **Weekly Progress Report for Week 4**

Dear Sir,
Following is the weekly progress report dating from 26th June to 2nd of July, 2023. I went ahead and researched on shearing machine and practiced on Datasets for a clearer understanding of the things.

My Understanding of the Project: INTP23-ML-5: Equipment Failure Prediction for Predictive Maintenance

Predictive Maintenance is the procedure of using already existing data of various factors which might cause equipment failure and using those data available to us to predict when an equipment might fail in the future. It basically works on the principle of Condition Monitoring. Condition-monitoring tools combined with artificial intelligence and machine learning techniques forecast expected machine failure.

Predictive maintenance helps in:

- reducing maintenance costs
- maintenance scheduling and planning
- improving reliability.

With the help of such technologies, we can predict and perform maintenance activities without disrupting normal machine activities.

Weekly Progress:

26th June 2023:
Practiced on Medine Price Dataset from Kaggle.

- Implemented numpy, pandas, Tensorflow, Keras.
- Used Data preprocessing and filteration to improve the data set.
- https://github.com/JehanPatel/Medicine-Price-Prediction

27th June 2023:
Practiced on Weather Prediction Dataset from Kaggle

- Implemented numpy, pandas, sklearn, Logistic Regression
- Implemented RandomForestClassifier and compared the different variables of the dataset.
- https://github.com/JehanPatel/Weather-Prediction-Dataset

28th June 2023:
Read on FSM Shearing Machine PDF.

- Implemented numpy, pandas, sklearn, Logistic Regression
- Implemented matplotlib functionalities to compare data variables!
- Had a few errors here and there but had it figured out later.

29th June 2023:
Practiced on Laptop Cleaning Dataset from Kaggle

- Implemented numpy, pandas, sklearn and seaborn to utilize the data properly.
- Implemented matplotlib functionalities to compare data variables.
- Utilized Random Forest, Gradient Boosting, XGBRegressor, CatBoosting Regressor and AdaBooster for the dataset.

30th June 2023:
Worked on the Largest University Dataset from Kaggle

- Implemented numpy, pandas, sklearn and seaborn to utilize the data properly.
- Implemented matplotlib functionalities to compare data variables.
- Utilized MultinobialNB functionalities for the dataset.

1st July 2023:
Learned on the basics of machine learning and the various models behind it.

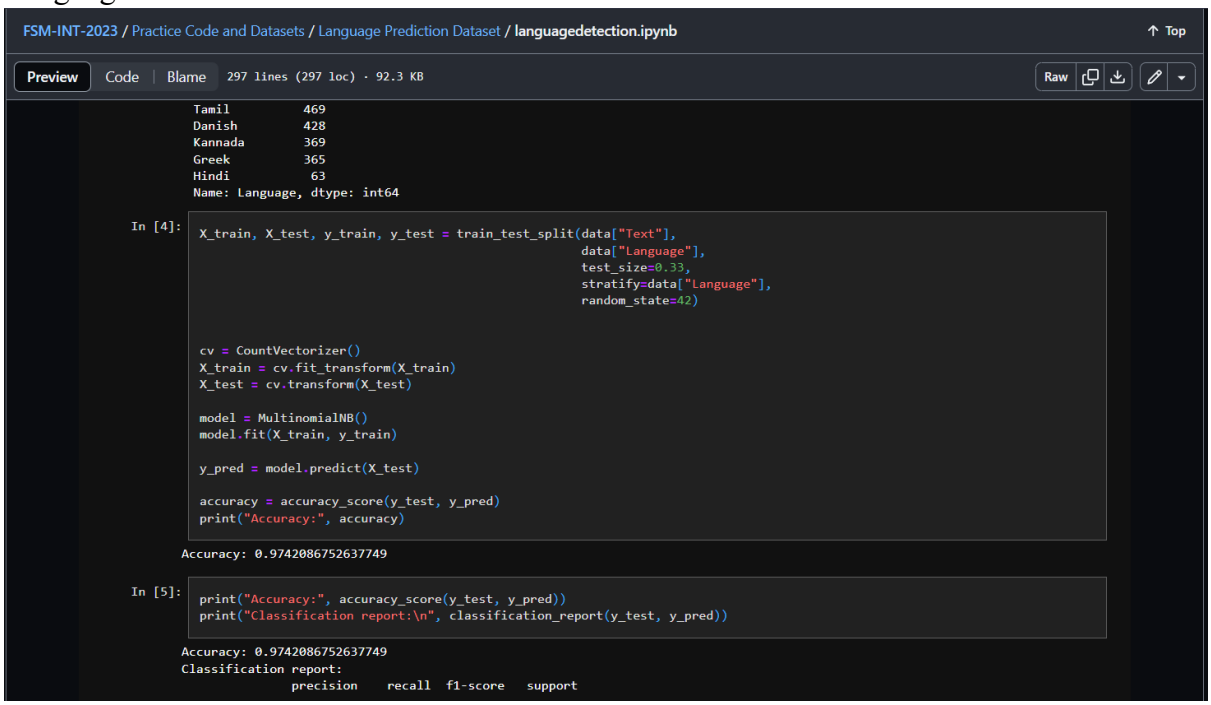
- Implemented numpy, pandas, sklearn and seaborn to utilize the data properly.
- Implemented matplotlib functionalities to compare data variables.
- Utilized xgboost and optuna for the dataset.

2nd July 2023:
Practiced on Flight Prediction Dataset from Kaggle

- Implemented numpy, pandas, sklearn and seaborn to utilize the data properly.
- Implemented matplotlib functionalities to compare data variables.
- Utilized seaborn functionalities for creating comparative data charts.

GitHub Repository: https://github.com/JehanPatel/FSM-INT-2023

Language Detection



FSM-INT-2023 / Practice Code and Datasets / Language Prediction Dataset / **languagedetection.ipynb** ↑ Top

Preview | Code | Blame 297 lines (297 loc) · 92.3 KB Raw

```
Tamil          469
Danish         428
Kannada        369
Greek          365
Hindi           63
Name: Language, dtype: int64
```

In [4]:

```
X_train, X_test, y_train, y_test = train_test_split(data["Text"],
                                                    data["Language"],
                                                    test_size=0.33,
                                                    stratify=data["Language"],
                                                    random_state=42)


cv = CountVectorizer()
X_train = cv.fit_transform(X_train)
X_test = cv.transform(X_test)

model = MultinomialNB()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

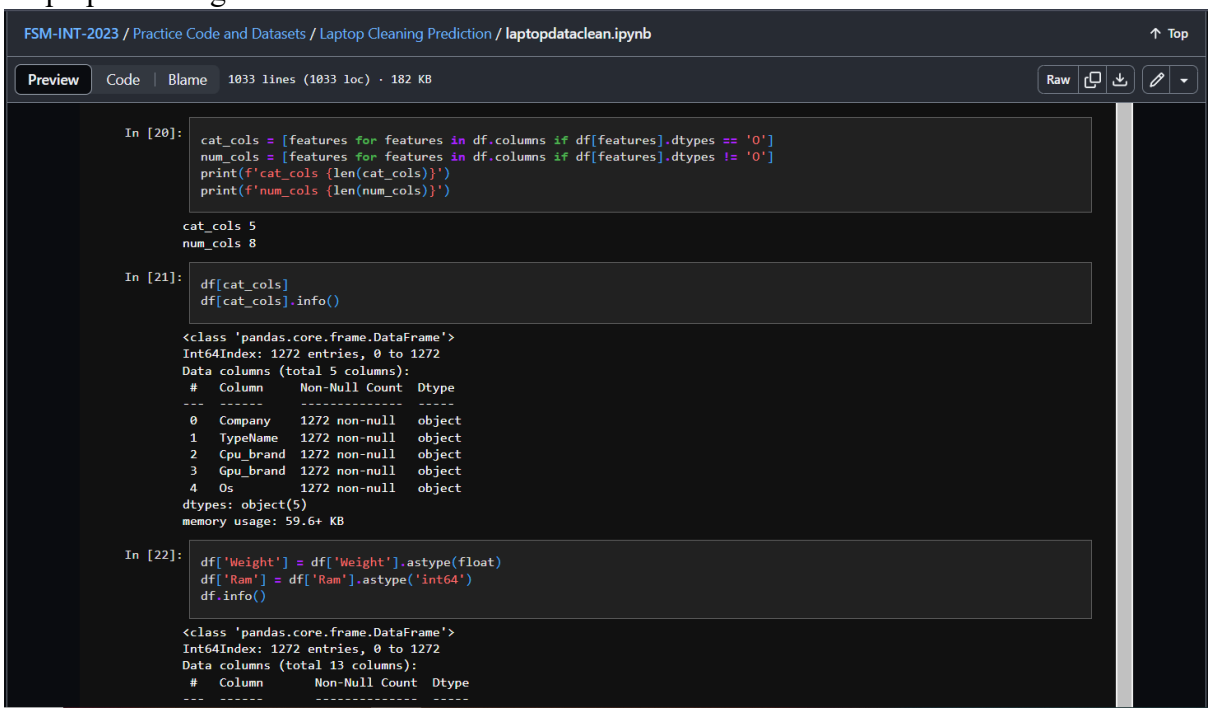
```
Accuracy: 0.9742086752637749
```

In [5]:

```
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.9742086752637749
Classification report:
              precision    recall  f1-score   support
```

Laptop Cleaning



FSM-INT-2023 / Practice Code and Datasets / Laptop Cleaning Prediction / **laptopdataclean.ipynb** ↑ Top

Preview | Code | Blame 1033 lines (1033 loc) · 182 KB Raw

In [20]:

```
cat_cols = [features for features in df.columns if df[features].dtypes == 'O']
num_cols = [features for features in df.columns if df[features].dtypes != 'O']
print(f'cat_cols {len(cat_cols)}')
print(f'num_cols {len(num_cols)}')
```

```
cat_cols 5
num_cols 8
```

In [21]:

```
df[cat_cols]
df[cat_cols].info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1272 entries, 0 to 1272
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   Company    1272 non-null   object
 1   TypeName   1272 non-null   object
 2   Cpu_brand  1272 non-null   object
 3   Gpu_brand  1272 non-null   object
 4   Os         1272 non-null   object
dtypes: object(5)
memory usage: 59.6+ KB
```

In [22]:

```
df['Weight'] = df['Weight'].astype(float)
df['Ram'] = df['Ram'].astype('int64')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1272 entries, 0 to 1272
Data columns (total 13 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
```

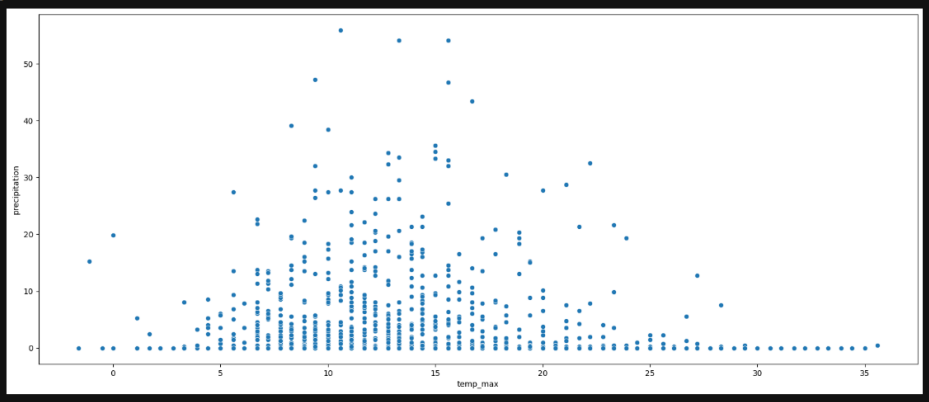
Weather Prediction

FSM-INT-2023 / Practice Code and Datasets / Weather Prediction / **weatherprediction.ipynb** ↑ Top

Preview | Code | Blame 677 lines (677 loc) · 606 KB Raw

In [6]:
```
fig, ax = plt.subplots(figsize=(20, 8))
sns.scatterplot(x='temp_max',y='precipitation',data=df,ax=ax)
```

Out[6]: <AxesSubplot:xlabel='temp_max', ylabel='precipitation'>



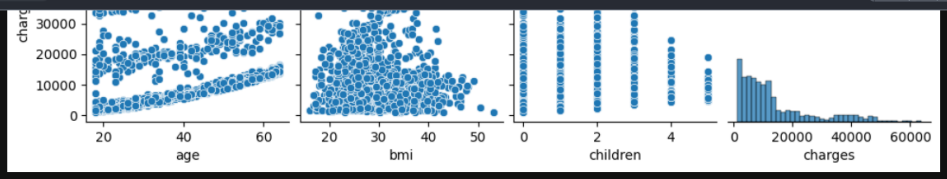
In [7]:
```
fig, ax = plt.subplots(figsize=(20, 8))
sns.scatterplot(x='temp_min',y='precipitation',data=df,ax=ax)
```

Out[7]: <AxesSubplot:xlabel='temp_min', ylabel='precipitation'>

Medical Price

FSM-INT-2023 / Practice Code and Datasets / Medical Insurance / **medicalprice.ipynb** ↑ Top

Preview | Code | Blame 781 lines (781 loc) · 389 KB Raw



In [10]:
```
def plot(df):
    fig, axes = plt.subplots(1, 3, figsize=(15, 5))
    fig.suptitle('Categorical features vs Count')

    # Sex
    sns.countplot(ax=axes[0], x="sex", data=medical)
    axes[0].set_title("Sex")
    axes[0].tick_params(axis='x', rotation=45)

    # Smoker
    sns.countplot(ax=axes[1], x="smoker", data=medical)
    axes[1].set_title("Smoker")
    axes[1].tick_params(axis='x', rotation=45)

    # Region
    sns.countplot(ax=axes[2], x="region", data=medical)
    axes[2].set_title("Region")
    axes[2].tick_params(axis='x', rotation=45)
plot(medical)
```