ChatGPT

AWS SageMaker Notebook Coding Assessment Cheatsheet

---

1. Seaborn Plots (with Syntax)

```
import seaborn as sns
import matplotlib.pyplot as plt

# Basic plots
sns.histplot(df['column'])  # Histogram
sns.boxplot(x='column', data=df)  # Boxplot
sns.countplot(x='column', data=df)  # Countplot (Categorical)
sns.barplot(x='col1', y='col2', data=df)  # Barplot (Aggregated)
sns.scatterplot(x='col1', y='col2', data=df)  # Scatterplot
sns.pairplot(df)  # Pairwise relationship plot
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')  # Correlation Heatmap
```

---

2. Matplotlib Plots (with Syntax)

```
import matplotlib.pyplot as plt

plt.plot(x, y)  # Line plot
plt.scatter(x, y)  # Scatter plot
plt.bar(x, y)  # Bar chart
plt.hist(data)  # Histogram
plt.boxplot(data)  # Box plot
plt.pie(values, labels=labels)  # Pie chart

# Labels and Titles
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.title('Title')
plt.legend()
plt.show()
```

---

3. Major Sklearn Libraries (For End-to-End ML Flow)

```
# Data Preprocessing
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
```

```
# Feature Selection
from sklearn.feature_selection import SelectKBest, chi2

# Model Selection and Tuning
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV,
cross_val_score

# Metrics
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report, roc_auc_score
```

---

4. Major Sklearn Models and Their Libraries

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,
GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import AdaBoostClassifier
```

---

5. SageMaker-Specific Code Snippets

a. Get Execution Role

```
import sagemaker
role = sagemaker.get_execution_role()
```

b. Read File from S3

```
import pandas as pd
s3_path = 's3://your-bucket/path/to/data.csv'
df = pd.read_csv(s3_path)
```

c. Save Trained Model to S3 (using `joblib` or `pickle`)

```
import joblib
joblib.dump(model, 'model.pkl')

import boto3
```

```
s3 = boto3.client('s3')
s3.upload_file('model.pkl', 'your-bucket-name', 'model-path/model.pkl')
```

d. Load Model from S3 in Notebook

```
s3.download_file('your-bucket-name', 'model-path/model.pkl', 'model.pkl')
model = joblib.load('model.pkl')
```

e. Pickling a Model

```
import pickle
with open('model.pkl', 'wb') as f:
    pickle.dump(model, f)
```

---

6. Start a Model Endpoint in SageMaker (Simple Form)

```
from sagemaker.sklearn.model import SKLearnModel

model = SKLearnModel(model_data='s3://your-bucket/model-path/model.pkl',
                     role=role,
                     entry_point='inference.py')

predictor = model.deploy(instance_type='ml.m5.large',
initial_instance_count=1)

# To make prediction
predictor.predict([[1, 2, 3, 4]])
```

`inference.py` should have a `model_fn`, `predict_fn`, and `input_fn` implemented for custom logic.

Tip: Always test your model pipeline locally first, then move it to SageMaker for deployment and scaling.