Sudhanshu Agarwal S1618 Cheat Sheet

Pandas => Allows us to analyse big data & make conclusions based on statistical theories

Essential operations =>

① Read Data → csv, excel
② Display Data → head()
③ => Indexing: df['col'], loc[]
④ => Manip → Filter, sort, Group
⑤ => Cleaning: Drop, Rename
⑥ => Viz => Basic plots

Numpy => Aims to provide an array obj 50x faster than traditional python lists.

Essential ops =>

① => Array Creation → numpy.array()
② => numpy.sum(), mean(), max()
③ => numpy.reshape(), concatenate()
④ => numpy.sqrt(), power(), abs()
⑤ => numpy.mean(), median(), std(), var()

Seaborn => Based on matplotlib

- Scatter plot => sns.scatterplot()
- Line plot => sns.lineplot()
- Bar plot => sns.barplot
- Box plot => sns.boxplot.
- Etc => pairplot, lgplot, violinplot

Matplotlib => used for creating static, animated & interactive visualizations.

Line plot => plt.plot(x,y)
Bar plot
Pie chart

Subplots =>
plt.subplot(row, cols, index)

Multiplot => (rows, cols)

Correlation =>
↳ coefficients measure from -1 to 1.

Types of corr.

- +ve corr
- -ve corr

strength
↳ closer to 1

Spacy (Day 7) => open source NLP library, key features =>

① Tokenization tokens = nlp(text)
② POS tagging: It can assign grammatical parts of speech
③ => NER => Spacy can identify & classify named entities in text, such as names of people etc.

for ent in doc.ents:
Print → ent text

Stop words => words ignored/filtered out during the processing of NL text because they are of less help. Eg => "the", "and" etc.

Word embeddings => Type of rep of words in a vector space, where words with similar meaning are mapped to similar vectors. Eg => word = "example"
embedding = nlp(word).vector

Sentiment Analysis with NLTK => for finding sentiment => included in NLTK

x

Regression => $Y = b_0 + b_1 X + E$ → constant (error)
Y = Dep. Var, X = indep. Var

Residuals => The diff b/w observed & predicted values.

Goodness of fit => R^2

OLS => model = smf.ols(formula = formula, df).fit()

x

Pivot table => pivot_df = pd.pivot_table()

Crosstab => pd.crosstab()

Regex => import re
patter = r"your_reg-ex"

Common patters =>
\d => Matches any digit
\w => Matches any alphanum char
\s => Matches any whitespace char

*Features =>
↳ Types of features =>
① => Numerical features
② => Categorical features

*Feature Engineering =>
↳ Techniques =>
① => Normalization
② => Scaling
③ => One-hot encoding

*Feature selection =>
↳ Labels
↳ Binary, multiclass & regression lables

*Eval metrics
↳ ① Binary Classification
↳ Accuracy, Precision, Recall, F1 score.
② → Multiclass, Regression classification

*Train-test split =>
↳ split ratio → 70-30 or 80-20
↳ Implementation =>
[*Random splitting
*Stratified splitting]

*Supervised learning
↳ Task => Predict or classify based on labled training data
Types of Algo =>
SVM, Decision trees, Neural nets, metrics =>
Accuracy, precision, recall, F1 score. etc.

*Unsupervised learning
↳ Algos => K-means, PCA, Apriori Algorithm.

*Machine learning Pipeline =>
↳ systematic way to organize the ML workflow
eg =>
```
num_transformer = Pipeline(steps=[
    {"imputer", SimpleImputer(strategy="mean")),
    {"scaler", StandardScaler()}
    ]}
```
one_hot encoding => encoder = OneHotEncoder()
Standard Scaling => scaler = StandardScaler()
(Z-score normalization)

*Dimension Reduction
↳ reduce no: of features while preserving the essential features.

*PCA => Linear dimensionality reduction technique
↳
```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
```

*t-SNE => Non-linear dimensionality reduction
↳
```
from sklearn.manifold import TSNE
tsne = TSNE(n_components=2, perp=30)
X_tsne = tsne.fit_transform(X)
```

*Classification =>
↳ Selecting a classifier
- Nature of data
- Data size
- Interpretability
- Computational resources

*Simple Classifiers => Logistic Reg, K-NN

*Dask =>
↳ Key features
- Parallelization
- Dynamic Task Scheduling
- Familiar API to numpy & pandas.

*Spark => open source distributed computing system.
→ In-memory processing
→ Distributed Computing
→ Resilient Distributed Datasets
→ Spark SQL.

*Advantages
→ Speed
→ Ease of use
→ Flexibility