Study Notes To Ace Your Data Science Interview

Preparing for interviews as a data science/machine learning intern, I realized there was a lot of knowledge to cover for the different rounds of the interview process. In order not to be overwhelmed with information overload, I decided to put together a document of the major focus points. This is the document I'll be sharing with you in this article.

My study notes were divided into twelve focus areas:

- Data analysis implementation
- Machine learning and deep learning algorithms
- Building and evaluating models
- Machine Learning (ML) theory
- Programming
- Statistics and probability
- Natural Language Processing (NLP)
- Recommendation Systems
- Project-based questions
- Behavioral questions
- Questions to ask your interviewer
- Questions to ask your recruiter

Data analysis implementation

Here, the focus was on syntaxes, methods, and processes for wrangling data.

1. **Exploratory Data Analysis (EDA):**
    a. Using methods such as df.head(), df.shape, and df.describe for EDA of a data frame.
    b. Other important analyses are getting unique values in a column or the count of unique values, grouping by one or more columns, and getting aggregates.
2. **Handling missing values:**
    a. Either using df.dropna(axis=0) to drop rows with missing values or df.fillna() to fill in values. You could either fill in with the mean or median of values in that column.
    b. Filling missing values with the mean does not work well with a diverse dataset/column that is skewed and has outliers. In this case, the median is more appropriate.
    c. If data was collected in some sort of other, bfill might also be a good method to use. Only fill missing data where data was not recorded, not because it does not exist. That should be kept as NaN.
3. **Data cleaning:**
    a. Pandas methods for dropping columns, renaming columns, changing the data types of columns, and resetting the index.
    b. Subsetting a data frame based on the specific numbers, column names, or condition(s), how to merge, join, or concatenate data frames on either row or column axis.

Machine learning and deep learning algorithms

You should be able to provide an in-depth explanation and also give reasons for whatever answers you give to most questions. I'll be sharing examples of areas you should consider focusing on and some questions you should think about.

1. **Machine learning algorithms:**
    a. **Neural networks (NN):**
        i. Why do you scale the features?
        ii. What is batch normalization?
        iii. How do you choose layers to add?
    b. **Support Vector Machines:**
        i. What are support vectors?
        ii. What are kernel functions and the types of kernel functions?
    c. **Linear and logistic regression**:
        i. Difference between linear and logistic regression.
        ii. Handling bias in these algorithms.
        iii. What are solvers for logistic regression?
    d. **Naïve Bayes**:
        i. Why is it naïve?
        ii. When does it perform really well?
2. **Compare machine learning algorithms**:
    a. Classification vs regression algorithms.
    b. Supervised vs unsupervised algorithms.
    c. How do different factors affect these algorithms, e.g. decision boundaries, training data size, and speed of training?
    d. Different hyper-parameters for these algorithms.
    e. When to use which algorithm, and why (give different use cases and why they have advantages there)?
    f. The loss functions for different algorithms.
    g. Which algorithms require scaled data for the algorithms and why do they need this?
    h. Difference between KNN vs K-means clustering
3. **Cross-validation**:
    a. What is cross-validation and why is it used in machine learning?
    b. When is cross-validation not necessary?
4. **Normalizing vs scaling vs standardization**: The difference between these three terms, knowing when to use each of them and why you should choose which.
5. **Categorical encoding vs target encoding vs ordinal encoding**:
    a. Definition of these encodings and differences between each.
    b. Use cases and examples where they could be applied.
6. **Generative and discriminative models:**
    a. Explanation of these two types of models and their differences.
    b. Examples of machine learning algorithms under each of these categories.
    c. Use cases of when one of the categories works better than the other.

7. **Types of outputs in classification problems:**
    a. Class outputs vs probability outputs, and how they work.
    b. What models give what types of outputs?
8. **Tree-based algorithms**:
    a. Which algorithms are built on the tree-based model?
    b. What kind of data do tree-based models have an advantage on (sparse vs dense data)?
9. **Ensemble models:**
    a. How they work.
    b. Boosting vs bagging.
    c. Advantages of different ensemble methods, preferred data types, and example algorithms based on each method.
    d. RandomForest vs GradientBoosting.

Building and evaluating models

1. **How to create features**: Different techniques that can be used to create features from a data set.
2. **Build a simple model**:
    a. Fit a model using a machine learning algorithm from scikit-learn.
    b. Predict the results on unseen test data.
3. **Tune hyperparameters**: What hyperparameters are available for the selected algorithm and what do they do? Understand the different methods of tuning hyperparameters and the pros and cons of these different cross-validation techniques.
4. **Validating a model**: Using a validation dataset.
5. **Feature engineering**: What it is, how to do it, and why it is important.
6. **Confusion matrix**:
    a. Using absolute numbers vs percentages.
    b. Calculating true positive rate, false positive rate, true negative rate, and false negative rate.
7. **Evaluation metrics for regression vs classification models**:
    a. Mean Absolute error (MAE), Root Mean Square Error (RMSE), R^2.
    b. Accuracy, precision, recall.
    c. Negative predictive value, specificity.
    d. F1-score.
    e. ROC, AUC, ROC-AUC.
    f. Diversity, coverage, serendipity, novelty, etc.
    g. When is accuracy not a good evaluation metric for a classification problem?
8. **Overfitting and underfitting**:
    a. The definitions of both theoretically and giving real-world examples.
    b. How to solve both problems.
    c. Bias-variance tradeoff and model complexity.
9. **Measuring feature importance.**

ML Theory

1. **Regularization**:
    a. Lasso regularization.
    b. Ridge regularization.
    c. Elastic net regularization.
    d. Comparison of L1 vs L2 regularization, the way they work, and when to use one over the other.
2. **Loss functions, cost functions, objective functions**:
    a. Definitions of each of these with examples.
    b. The problems these functions solve.
    c. How do they relate to one another?
3. **Entropy:**
    a. Explain entropy in machine learning.
    b. What machine learning algorithms use entropy and how is it applied in the algorithms?
4. **How do you deal with outliers?**
5. **How do you handle imbalanced datasets in a classification model?**
6. **How can you avoid overfitting?**

Programming

1. **Time complexity and efficiency of Python sort and other inbuilt methods.**
2. **Data structures and algorithms**.
    a. Which data structures or algorithms to apply to solve a problem optimally.
    b. Writing code for some data structures and algorithms from scratch, and not using inbuilt methods.
3. **Big-O notation**:
    a. Space and time complexity for different data structures and algorithms.
    b. Which to optimize for given a specific problem and limitations.

Statistics and probability

1. **Statistical distributions**:
    a. Normal, uniform, binomial, poisson, bernoulil distributions.
    b. Multinomial, multinoulli, uniform distributions.
    c. The behaviors, properties, basic calculations, and use cases of each of these distributions.
2. **Sampling**:
    a. Population sampling.
    b. Central limit theorem.
3. **Random variables**:
    a. Discrete variables .
    b. Continuous variables.
4. **Statistical analysis**:
    a. Variance.
    b. Standard deviation.

    c. Covariance.
    d. Correlation.
    e. Regression.
5. **Probability theory**:
    a. Properties of probability.
    b. Probability distributions (probability density function, probability mass function, cumulative density function).
    c. De Morgan's law.
6. **Probability events**:
    a. Independent/mutually exclusive events.
    b. Non-mutually exclusive events
    c. Disjoint events.
    d. How are these all different or similar (overlapping)?
7. **Hypothesis and A/B testing**:
    a. Statistical significance.
    b. Null and alternative hypotheses, with examples.
    c. Type I and II errors.
    d. What are the p-value, statistical power, and confidence level, and how are they calculated?
8. **Bayes' rule**:
    a. Definition of Bayes' rule.
    b. Formula to calculate conditional probability based on this rule.
    c. Application of Bayes' rule in real-world probability calculations.

Natural Language Processing (NLP)

1. **Definitions**:
    a. Vocabulary.
    b. Language model.
2. **Text pre-processing/analysis**:
    a. Stemming.
    b. Lemmatization.
    c. Tokenization.
    d. Stop words.
    e. TF-IDF.
3. **Text vectorization**:
    a. One-hot encoding
    b. Bag of Words
    c. Word embeddings/word vectors
    d. Sub-words.
4. **Model architectures**:
    a. Recurrent Neural Networks (RNNs).
    b. Long Short-Term Memory (LSTM).
    c. Transformers (self-attention).
    d. Pros and cons of each of them.

Recommendation Systems

1. **Types of recommendation systems**:
    a. Content-based
    b. Collaborative filtering
    c. Knowledge-based
    d. Hybrid recommender systems.
    e. Pros and cons of each, use-cases, and when they will not perform well.
    f. Explain the cold-start problem in collaborative filtering, when it could occur and how to fix it.
2. **Ranking and clustering algorithms**.
3. **Performance evaluation for recommender systems**.
4. **What to consider and optimize for when designing a recommender system**: Accuracy, relevance, speed, latency, diversity.

Project-based questions

1. Explain your ML project process?
2. What's your favorite algorithm, and can you explain it in less than a minute?
3. What are your favorite use cases of machine learning models?
4. Specific questions about a project on your resume or portfolio: You should be able to talk about the end-to-end process, from the business needs, planning process, data collection, building the model, evaluating performance, deploying, and measuring performance in production.

Behavioral questions: It is very helpful to use the STAR (Situation, Task, Action, quantifiable Result) framework in answering these questions. Also, try to make it personal and take more responsibility for your work and contributions, by saying more of I, than we.

Here are some sample questions to consider while prepping:

1. Tell me about yourself: Talk about your background. Describe your interests. Mention your experience. Explain why you're excited about the opportunity.
2. Why do you want to work with the company?
3. What do you think is the most valuable data in the business?
4. What has been the most significant accomplishment in my career so far / biggest success?
5. Where do you want to take your career? What do you want next?
6. Describe the last time you had to adjust course to more effectively reach a goal? What was the goal, how did you adjust, and what was the outcome?
7. Would you prioritize speed of delivery or quality of the end product?
8. Talk about a project that you worked on that failed.
9. Talk about a time when you didn't think you could do something.
10. Talk about a time when the people around you disagreed about something, and how you resolved it.

Questions to ask your interviewer

1. Ask about the day-to-day responsibilities of the role for which you have been interviewed.
2. What makes the best intern (new hire) on their team stand out / what are the key values and characteristics that they look for in an employee?
3. What are the metrics on which your performance will be evaluated while working in the company / what are the expectations for this role?
4. Is there anything about your experience or skills that they have reservations about? If so, let them know that you would like to address their concerns.
5. What is the typical career path for someone hired for this role?
6. What is their favorite part about working in this company and what is the most challenging aspect of this job?

Questions to ask your recruiter

1. How many steps are in the interview process and what are the next steps currently?
2. Is it clear yet what team you will be placed in?
3. Is there room to switch teams or shadow people on other teams?
4. Is there relocation or accommodation assistance?
5. How long have they been with the company and why do they like working there?
6. What did they think about the company before they joined and what do they think now?

Conclusion

I hope this is helpful to you, not as a comprehensive syllabus, but as a guide to the various topics, you could study for your interviews. Data science is a very broad field and the role differs greatly by company. It is important to do in-depth research on the specific company, read the job description carefully, and speak to your recruiter to get an idea of what focus areas to lean more towards.

I also understand that the job application and interview process can be pretty daunting. If you are faced with rejections, please do not lose track of the fact that it is not necessarily a reflection of your knowledge gap, nor is it a measure of your worth as a human being, or a measure of your intelligence. Remember to give yourself grace, ask for feedback, make improvements in whatever way you can, and keep practicing and sending in those applications.

Please let me know what you think about this piece and kindly share this with anyone you think might benefit from this. I would also appreciate your suggestions on any other topic you might like me to write about in relation to job applications and getting data science roles.

Thank you for reading.

Aniekan.