IBM Data Science Professional Certificate Notes

Contents

Data Science Methodology

From Problem to Approach

1) What is the problem that you are trying to solve?
2) How can you use data to answer the question?

Working with The Data

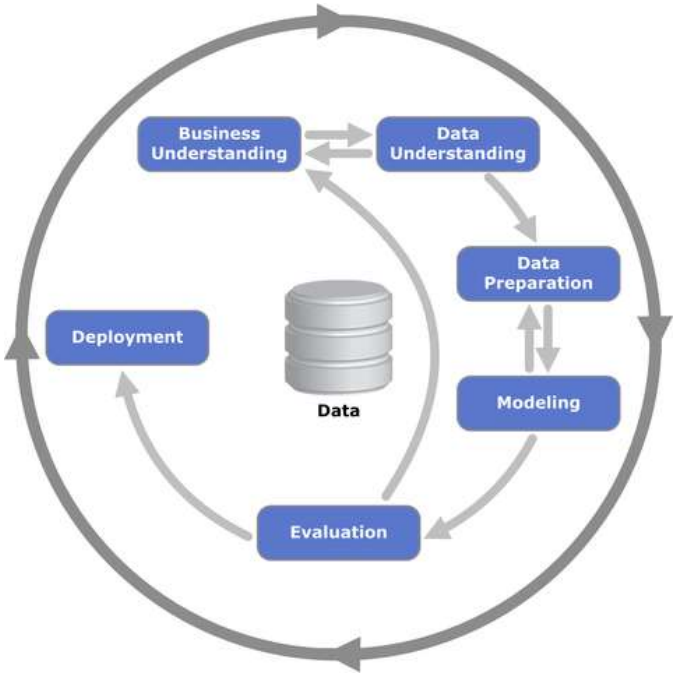
3) What data do you need to answer the question?
4) Where is the data coming from (identify all sources) and how will you get it?
5) Is the data that you collected representative of the problem to be solved?
6) What additional work is required to manipulate and work with the data?

Deriving The Answer

7) In what way can the data be visualised to get to the answer that is required?
8) Does the model used really answer the initial question or does it need to be adjusted?
9) Can you put the model into practice?
10) Can you get constructive feedback into answering the question?

CRISP-DM

- Cross Industry Process for Data Mining
- Business understanding
- Data understanding
- Data preparation
- Modelling
- Evaluation
- Deployment



The CRISP-DM Methodology.

Business Understanding

- What is the problem you are trying to solve?
- It is important to establish the exact goal, so you know what you want to solve

- Once you have the goal you need to split it into objectives and requirements that can be met
- Different stakeholders can be incorporated into deciding the objectives

Types of Models

- If the question is to determine probabilities use a predictive model
- If the question is to show relationships use a descriptive model
- If the question has a yes/no answer use a classification model

Data Requirements and Collection

- Identify necessary data content, formats, types, properties and collection methods
- It is important to think about the future stages and how the collected data can affect these
- How much data is needed?
- What quality of data is needed? This is be visualised
- How easy is it to collect the proposed data?
- It is alright to defer decisions about some unavailable data, and attempt to collect it at a later stage

Data Understanding and Preparation

- Can use descriptive statistics such as:
    - Univariate statistics
    - Pairwise correlations
    - Histograms
- Highly correlated features mean that one may be redundant in a model
- Univariate statistics or histograms can be used to assess data quality
- Missing values will need to be filled in, or the feature dropped
- Invalid or missing values must also be fixed

Modelling and Evaluation

- Modelling and evaluation occur iteratively
- Develop models that are descriptive or predictive
- Statistically or machine learning driven
- Training set, historical data with known answers, is used to create the model
- Evaluation decides whether the model built meets the initial requirements

Deployment

- Stakeholders must be familiar with the tool for it to work
- It can be rolled out to limited users or in a test environment first to make sure it works
- Get feedback from users, is the problem solved?
- Should have a review after 3-12 months to see if the model needs refining

SQL for Data Scientists

Select

- In it's simplest form, retrieving data is done with:
  SELECT [column] FROM [table] WHERE [clause]

String Patterns

- String patterns can be used to search for data
- WHERE name LIKE 'R%' would get names beginning with R
- The % acts as a wildcard of zero or more characters
- The _ acts as a wildcard of exactly one character
- WHERE number BETWEEN 100 AND 200 gets numerical values between 100 and 200 inclusive
- WHERE name IN ('Bob', 'Carly') gets values that are within the given list

Sorting Results

- The result set can be ordered using the ORDER BY clause
- Data can be sorted in ASC or DESC order
- Data is sorted in ascending if not specified

Distinct Results

- To eliminate duplicates, use DISTINCT
- e.g. SELECT DISTINCT(country) FROM authors

Grouping Results

- Data can be grouped using the GROUP BY clause
- Aggregate functions can then be used on these groups
- Aggregate functions include COUNT, SUM, MIN, MAX, AVG
- HAVING can be used with the GROUP BY clause, as a WHERE for aggregate functions
- e.g. SELECT country, COUNT(country) FROM authors GROUP BY country HAVING COUNT(country) > 5

Built-In Functions

- Most databases come with built-in SQL functions
- These functions can be included as part of SQL statements
- Aggregate functions are an example of built-in functions
- Scaler and string functions act on individual values
- These include ROUND, LENGTH, UCASE, LCASE
- Scaler functions can also be used in the WHERE clause, such as WHERE LCASE(animal) = 'cat'
- Scaler functions can also be used within other functions, such as DISTINCT(UCASE(animal))

Dates and Times

- Special types for dates and times exist:
  - DATE (YYYYMMDD)
  - TIME (HHMMSS)
  - TIMESTAMP (YYYYXXDDHHMMSSZZZZZZ)
- Functions such as DAY, MONTH, YEAR, HOUR, MINUTE, DAYOFWEEK exist to operate on these types
- Arithmetic can also be performed on these types

- SALE_DATE + 3 DAYS
- RETURN_DATE - SALE_DATE

DB-API

- DB-API provides a common interface to relational databases in Python
- Some applications or databases also have their own APIs
- Connection objects create database connections and manage transactions
- A cursor object allows access to database records and result sets

```
from dbmodule import connect

# Create connection object
connection = connect('dbname', 'username', 'pass')

# Create cursor object
cursor = connection.cursor()

# Run queries
cursor.execute('SELECT * FROM mytable')
results = cursor.fetchall()

# Free resources
cursor.close()
connection.close()
```

Data Analysis in Python

Important Data

- Pandas can be used to import data from files to dataframes
- Depending on the file type there are different methods to read in data:
    - read_csv()
    - read_json()
    - read_excel()
    - read_sql()
- To make sure the data is in correctly, a protion can be viewed using df.head()
- The data can also be described using df.describe()

Missing Values

- Missing data is typically represented using NaN (not a number), ?, or 0
- Check with data collection source to see if they have the missing data
- Dropping the missing values is always an option
- The best option is to replace the missing values
- Mean, median, and mode are simple ways to replace the values
- More complex functions can be used to replace missing values
- df.dropna(axis = 0) drops rows with NaN
- df.dropna(axis = 1) drops columns with NaN
- df.replace(np.nan, new_value) can be used to replace missing values

Data Formatting

- Data is often from different sources compiled by different people
- Data formatting is the part of data cleaning that gets data into the same format
- e.g. 'cat' may be the same as 'kitten', 'kitty', and 'feline', or 1m is equal to 100cm or 1000mm
- Data types may also need changing
- df.dtypes() can be used to check the data types of values/columns
- df.astype('float') can be used to change to data type of values/columns

Data Normalisation

- Normalisation gets different data into a common scale/range
- 0 to 1 is a common normalisation range
- Simple feature scaling divides each value by the maximum
- Min-Max subtracts the minimum value and divides by the range
- Z-score (standard score) substracts the mean and divides by the standard deviation
- Z-scores are usually between -3 and 3

Binning in Preprocessing

- Grouping values into bins
- This converts numerical values into categorical values
- To convert pricing into three categories:
  bins = np.linspace(min(df.price), max(df.price), 4)
  group_names = ['Low', 'Medium', 'High']
  df['price-binned'] = pd.cut(df.price, bins, labels = group_names, include_lowest = True)

Categorical to Quantitative

- Most statistical models cannot take objects or strings as inputs

- One-hot encoding creates a new feature for possible category, and sets them to 0 or 1 if the data point has that category
- pd.get_dummies(column) performs one-hot encoding in Python

Exploratory Data Analysis (EDA)

- Summarises main characteristics of the data
- Gain a better understanding of the data
- Uncover relationships between variables
- Extract important variables

Descriptive Statistics

- Describes basic features of the data set
- This can be done simply using df.describe()
- Simple statistics include:
    - Count
    - Mean
    - Standard deviation
    - Min
    - 25% quartile
    - 50% quartile
    - 75% quartile
    - Max
- df.value_counts() summarises categorical data

Grouping Data

- df.groupby() groups data in a data set together
- Data can be grouped by multiple features if wanted
- Aggregations can then be applied to groups
- df.pivot(index = feature_1, columns = feature_2) can be used to put one feature as rows, and one as columns
- These pivots can be put into a heatmap using the following matplotlib code:
    - plt.pcolor(df_pivot)
    - plt.colorbar()
    - plt.show()

Correlation

- Measures to what extent different variables are interdependent
- Correlation doesn't imply causation
- A scatter plot or regression plot can be used to see correlation between two variables
- A positive correlation is when one variable increases, the other increases too
- A negative correlation is when one variable increases, the other decreases
- Pearson correlation can be used to measure correlation between two varaibles
- Pearson correlation gives a correlation coefficient (r) between -1 and 1 and a p-value between 0 and 1

Analysis of Variance (ANOVA)

- Statistical comparison of groups
- Can be used to find correlation between groups of a categorical variable
- ANOVA returns an f-test (variation between sample group means divided by variation within sample groups) and a p-value (confidence degree)

Linear Regression

- Using the input features we can fit a line to the target feature
- This line can be used to make new predictions
- LinearRegression can be used in the sklearn package
- Multiple linear regression refers to instances where you use multiple different features/parameters

Pipelines

- Pipelines can be used to simplify the model process
- pipe = Pipeline([('scale', StandardScaler()), ('polynomial', PolynomialFeatures(degree=2)), ('model', LinearRegression())])
- The pipeline has a list of tuples which contain fitable objects
- The pipeline must end in an object that can be used to make predictions

Model Evaluation

- The model should be evaluated on data it has never seen before
- To do this it can be trained on a training set, and evaluated on a test set
- The generalisation error/performance is how well the data performs on unseen data
- Cross validation gives a better understanding of the generalisation error, where the training set is split into several folds and each one is used as the test set in a different iteration

Overfitting and Underfitting

- Underfitting is when the model does not fit enough to the training data, possibly because there are not enough relevant features
- Overfitting is when the model fits to the training data too well so is fitting to noise as opposed to the natural trend
- A good model should neither underfit nor overfit

Grid Search

- Allows us to go through different model hyperparameters and choose the best combinations
- Hyperparameters are not calculated during fitting, and must be selected beforehand
- Grid search is exhaustive, and we select the combination with the lowest error
- model_selection.GridSearchCV() can be used for grid search in the sklearn package

Data Visualisation in Python

Area Plots
- df.plot(kind = 'area')
- Also known as an area chart or area graph
- Used to represent cumulated totals over time
- Based on the line plot

Histograms
- count, bin_edges = np.histogram(df[column])
- df[column].plot(kind = 'hist', xticks = bin_edges)
- Represent frequency distribution of a variable
- Vertical axis is frequency, horizontal axis has bins of values

Bar Charts
- df.plot(kind = 'bar')
- Compares the values of a variable at a given point in time
- The kind hbar can be used to make them horizontal

Pie Charts
- df.plot(kind = 'pie')
- Used to represent proportions of a whole
- It is often difficult to accurately represent data using pie charts

Box Plots
- df.plot(kind = 'box')
- Statistically represents data using minimum, first quartile, median, third quartile, and maximum
- Outliers are shown as individual dots of the box plots

Scatter Plots
- df.plot(kind = 'scatter', x = independent, y = dependent)
- Dependent variable to be plotted against the independent variable
- Can be used to show correlation between the two variables
- A bubble plot is a variation of the scatter plot that displays three dimensions of data

Waffle Charts
- Normally created to display progress toward goals
- Data in different groups are transformed into tiles which are fitted together
- Matplotlib does not have its own function for creating a waffle chart

Word Clouds
- Depiction of frequency of different words in some textual data
- Matplotlib does not have its own function for creating word clouds

Seaborn Regression Plots
- sns.regplot(x='year', y='total', data=df)
- Seaborn is based on matplotlib
- Seaborn greatly reduces the number of lines needed to create plots
- Additional parameters such as color and marker are accepted

World Maps

- Folium is a Python library that allows you to show geospatial data
- Any area can be plotted providing you know the latitude and longitude
- folium.Map() creates a world map
- These maps are also interactive
- folium.Map(location=[56.130, -106.35], zoom_start=4) creates a map centered around Canada
- tiles can be used to give different map representations (such as 'Stamen Toner' for visualising water or 'Stamen Terrain' for visualising hill shading and natural terrains)

Markers and Feature Groups on World Maps

- Feature groups are created using folium.map.FeatureGroup()
- Children can be added using the .add_child() function
- .add_child(folium.features.CircleMarker([51, -83], radius = 5, color = 'red')) adds a circle marker as a child
- The feature group can then be added to the map using .add_child() on the original map

Choropleth Maps

- Geospatial plot where areas are coloured in relative to some variable
- Typically, darker colours imply the variable is higher
- A geojson file is required to define to map for choropleths in Folium
- Maps have a choropleth function for creating the choropleths

Machine Learning in Python

Regression

- Can we predict one continuous value by using other numerical values
- The dependent variable is the target that we are trying to predict
- The independent variables are features in the model, that explain our prediction or the dependent variable
- Independent variables do not have to be continuous
- Simple regression uses only one independent variable (but does not have to be linear)
- Multiple regression uses many independent variables

Regression Algorithms

- Ordinal regression
- Poisson regression
- Fast forest quantile regression
- Linear, polynomial, lasso, stepwise, ridge regression
- Bayesian linear regression
- Neural network regression
- Decision forest regression
- Boosted decision forest regression
- KNN

Simple Linear Regression

- $y = a + bx$
- y is the dependent variable
- x is the independent variable
- a (intercept) and b (slope) are coefficients that must be calculated
- The line is fitted to minimise error on the training set
- Mean squared error is a common error to use in regression
- Linear regression is preferred because it is simple and fast
- Simple linear regression also does not require parameter tuning

Model Evaluation Approaches With Regression

- There are two main types of evaluation with regression:
    - Train and test on the same dataset
    - Train/test split
- K fold cross validation splits the training set into k different groups, so that train/test split can be performed k times

Evaluation Metrics With Regression

- Evaluation metrics are used to explain the performance of the model
- Common metrics with regression include:
    - Mean absolute error
    - Mean squared error
    - Root mean squared error
    - Coefficient of determination

Multiple Linear Regression

- Each independent variable has a separate coefficient that needs to be calculated
- However, coefficient calculation can be performed in the same way

Non-Linear Regression

- A scatter plot can be used to check whether two variables have a linear relationship
- This scatter plot can be used to determine whether a non-linear regression is needed, such as logarithmic, polynomial, or exponential
- By making independent variables polynomials (e.g. x^2, x^3) you can create the model in the same way you'd create a linear model

Classification Algorithms

- Decision trees
- Naive Bayes
- Linear discriminant analysis
- K-Nearest neighbour (kNN)
- Logistic regression
- Neural networks
- Support vector machines (SVM)

K-Nearest Neighbours

- If you are trying to predict the class of a point, you can find the nearest neighbour in the training set and use that class
- In kNN you can select k nearest neighbours, and select the most frequent class from them
- The nearest neighbours are chosen using a distance metric such as Manhattan or Euclidean distance
- Although it is difficult to visualise, kNN models can be trained in multi-dimensional space (many features)
- k is selected before fitting the model
- It is possible to use kNN for regression by taking the median or mean target value of neighbours

Evaluation Metrics in Classification

- A confusion matrix can be used to show true classes against predicted classes
- This can be used to see which classes the model gets confused between
- In a binary classifier it shows the true positives (TP), false negatives (FN), false positives (FP), and true negatives (TN)
- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F1-Score = 2 x (Pre x Rec) / (Pre + Rec)
- Log loss can be used to assess the performance of models which output probabilities between 0 and 1

Decision Trees

- A decision tree maps out all possible paths in the form of a tree
- The decision tree should split the training set into distinct nodes, with each node mostly containing one class

Logistic Regression

- Logistic regression is good when the data is binary
- Logistic regression is good when you need probabilistic results
- It will return a linear decision boundary (but by using polynomial features you can get other decision boundaries)
- The model will easily allow you to understand the importance of different features

Clustering

- Clustering creates a number of clusters, where each cluster contains similar data points
- This is an unsupervised learning method as there is no correct answer or known classes
- Clustering can be used to separate a dataset into different groups, known as clustering for segmentation
- Exploratory data analysis, summary generation, outlier detection, finding duplicates, and pre-processing are common uses for clustering

k-Means Clustering

- Divides data into non-overlapping subsets (clusters)
- Examples within a cluster are very similar
- Examples in different clusters are very different
- k-Means tries to minimise the intra-cluster distances and maximise the inter-cluster distances
- Different starting conditions can give different final results
- Therefore it is common to repeat the algorithm multiple times
- Mean distance of data points to cluster centroid can be used to evaluate the performance
- The "elbow point" on a graph of mean distance against k can show a good value for k

k-Means Algorithm

1) Initialise k centroids randomly
2) Assign each data point to the closest centroid
3) Recalculate the centroids as the mean of the cluster
4) Repeat until enough iterations or no changes to centroids

Hierarchical Clustering

- Hierarchical clustering creates a hierarchy of clusters
- Divisive is top-down, dividing clusters
- Agglomerative is bottom-up, pairing clusters together

Agglomerative Clustering

- A distance matrix between every point must be calculated
- The closest two points/clusters are merged together
- The new distance to the cluster for every other cluster can be calculated with a mean
- Every cluster is merged, one at a time, to create a hierarchy

Distance Between Clusters

- Single-linkage clustering uses the minimum distance between clusters
- Complete-linkage clustering uses the maximum distance between clusters
- Average linkage clustering uses the average distance between clusters
- Centroid linkage clustering uses the distance between cluster centroids

DBSCAN Clustering

- Density-Based Spatial Clustering of Applications with Noise
- A density-based clustering algorithm
- This can make arbitrary-shaped clusters
- Locates regions of high density and finds outliers
- Radius, R, defines space where enough points must be found in for in to be a dense area
- Minimum number of neighbours, M, is the minimum number of points to define a cluster
- DBSCAN finds a core point (with respect to R and M)
- Core points join to neighbours which join to other core points, these all form a cluster
- An outlier is neither a core point, nor is it near a core point

Recommender Systems

- Recommender systems are used to predict similar objects that people will like, compared to previously known items
- For example, Netflix will recommend new movies based upon what you have already watched, Facebook will recommend friends based upon your current profile
- They are used to keep users interested and provide a good user experience
- Content-based systems recommend similar items to what the user likes
- Collaborative filtering systems suggest items depending on what neighbours or other similar users like

Content-Based Recommender Systems

- The system will already know what a user likes, or what they have already interacted with
- One-hot encoding can be used, followed by averaging, to compare different categories
- However, there must be a way of dealing with new categories

Collaborative Filtering

- User-based collaborative filtering is based upon how similar two people are
- It can then suggest something that a similar user likes
- Item-based collaborative filtering, items build neighbourhoods based upon the user that have interacted with them (not the other way around
- Difficulties with collaborative filtering include data sparsity, cold start, and scalability