UDACITY

Return to Classroom

Communicate Data Findings

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Great work with this project, you did a great job of this using step by step processes during your exploratory process before moving to the explanatory process to create a clear and illustrative slideshow. Well done.

Your project included well written Python code and good visualisations.

This is the last and final project, I hope you have enjoyed this course.

All the best!

Below are some related courses which could be of interest.

- Data Science With R – Includes work with R, SQL, command line and git.
- Predictive Analytics for Business – Applying predictive analytics and business intelligence
- Intro to Machine Learning – Learning machine learning techniques with Pytorch.
- Data Scientist – Gain experience with real world data science projects.

Code Quality

All code is functional (i.e. no errors are thrown by the code). Warnings are okay, as long as they are not a result of poor coding practices.

All your **Python code** runs without error, well done.

Considering the focus of this course is not programming, you did a great job.

The project uses functions and loops where possible to reduce repetitive code. Comments and docstrings are used as needed to document code functionality.

You have used **functions** and **loops** where possible and **clear variable names** to increase code readability.

Going forward, try using **Python Docstring Comments** when writing functions where possible.

```
In [27]:
# plots a regression model
def draw_regression(xval,yval,x,cat,p,a,l,xt,yt,t):
    sb.lmplot( x=xval, y=yval, data=x, fit_reg=False, hue=cat, legend=False, palette=p, scatter_kws={"alpha":a,"s":15} )
    plt.legend(loc=l, markerscale=2)
    plt.xlabel(xt)
    plt.ylabel(yt)
    plt.title(t, loc='left')
    plt.show()
```

```
In [28]:
# filters the dataframe based on start time
def filter_dataframe(x,val1,val2):
    x_new = x[(x[st] >= val1 ) & (x[st] <= val2)]
    return x_new
```

Docstring Comments

Python has a built-in concept called docstrings, which is a great way to associate documentation you've written with Python modules, functions, classes, and methods. A docstring is added as a comment right below the function, module, or object head, and describes what the function, module, or object does. It is expected to follow these rules:

- A docstring is either a single line, or a multi-line comment. In the latter case, the first line is a short description, and after the first line an empty line follows.
- Begin the docstring with a capital letter, and end it with a period.

This is a basic example of what it looks like:

```
def add(value1, value2):
    """Calculate the sum of value1 and value2."""
    return value1 + value2
```

Here are some additional links for writing clean Python code.

- Real Python website includes tutorials and more.
- Earth Data Science website is more focused on Python code with earth data science.

Exploratory Data Analysis

The project (Parts I alone) contains at least 15 visualizations distributed over univariate, bivariate, and multivariate plots to explore many relationships in the data set. Reasoning is used to justify the flow of the exploration.

You have appropriately used **univariate**, **bivariate**, and **multivariate** plots where required. You produced a very good step by step process during your exploration.

When I first started this project I found this link Univariate, Bivariate, and Multivariate Analysis useful as it provides examples of when to use certain charts.

Questions and observations are placed regularly throughout the report, after each plot or set of related plots.

> Tip: Use the ""Question-Visualization-Observations"" framework throughout the exploration.
>
> Tip: For the Part I notebook, use *File > Download as... > HTML or PDF* menu option to generate the HTML/PDF.

You added **observations** and **questions** throughout the report explaining why areas were being explored. This is a great **analysis strategy** to use as it helps to discover answers or insights within the data.

Benefits

- Helps to discover **findings** and **answers** within large chunks of unreadable data.
- Can provide **logical steps** and **create guidance** on what to investigate next.
- Helps you keep track of what you have find as your analysis continues.

"Visualizations made in the project depict the data in an appropriate manner that allows plots to be readily interpreted. This includes choice of appropriate plot type, data encodings, transformations, and formatting (title, axis-labels) as needed.

> Tip: Do not overplot or incorrectly plot ordinal data."

You have used the appropriate visualisations at each stage of your analysis. This will help as you progress through your exploration of the data and ensures your investigation is accurate.

Using inappropriate visualizations could provide misleading answers.

Good effort with the scatter plots, you have applied **transparency** techniques to increase readability.

Going forward, try to also apply **jitter** techniques to increase the readability further.

Explanatory Data Analysis

The README.md must include a summary of main findings that reflects on the steps taken during the data exploration. It should also describes the key insights that are conveyed by the explanatory presentation.

> **Tip: The README.md summary is based on the exploration report (Part I notebook) and will guide your explanatory slide deck (Part II notebook) .**

You have provided a read-me file that contains information on the data set, a summary of the findings, and key insights for the presentation.

Benefits of a readme file.

- Ideally, if you add this project to GitHub the readme file will be the first file someone would see.
- The ReadMe file is important as it shows what you were investigating, how you achieved it and additional detail which was important in your analysis.

- **A slideshow (HTML file) is provided, with at least 3 visualizations, to convey key insights. Only selective plots are added to the slideshow from the exploratory analysis.**
- **The total number of visualizations in the slideshow is less than 50% of the number of visualizations in the exploratory analysis. For example, if the exploratory analysis (Part I) has 18 visualizations, the slideshow can have (3 - 8) visualizations.**
- **The key insights in the slideshow match those documented in the README.md summary.**
- **Each visualization in the slideshow is associated with comments that accurately depict their purpose and observation.**

> **Tip: For Part II notebook, use the `jupyter nbconvert` command to generate the HTML slide show.**

Excellent work with the slideshow.

The slideshow helps to highlight the main areas of your data exploration and whittle down the information to something you can easily explain and visualise to an audience, well done.

Why is this important

- This helps illustrate your findings after exploring data sets.
- Slideshows present your findings using clean, polished charts and key insights which are easy for audiences to read and digest.

All plots in the slideshow are appropriate, meaning the plot type, encodings, and transformations are suitable to the underlying data.

All plots in the slideshow are polished, meaning all plots have a title with labeled axes and legends. Labels include units as needed. In other words, each plot must have - chart title, x/y axis label (with units), x/y ticks, and legend.

Good effort with the visualisations which include clear readable chart titles, axis titles and labels where appropriate. This shows you can clearly create and present great visualisations for the audience to see and understand.

Suggestion

- You might find this link regarding do's and don'ts when creating charts useful:

DOWNLOAD PROJECT

RETURN TO PATH