Vizziest

https://www.ischool.berkeley.edu/projects/2019/vizziest-making-visualization-easiest-everyone

W210.6 Capstone Project

Chi Iong Ansjory, Jeff Braun, Grace Lin, Yang Yang Qian
[ansjory, jbraun, yqlin, yangyang.qian]@ischool.berkeley.edu

Vizziest

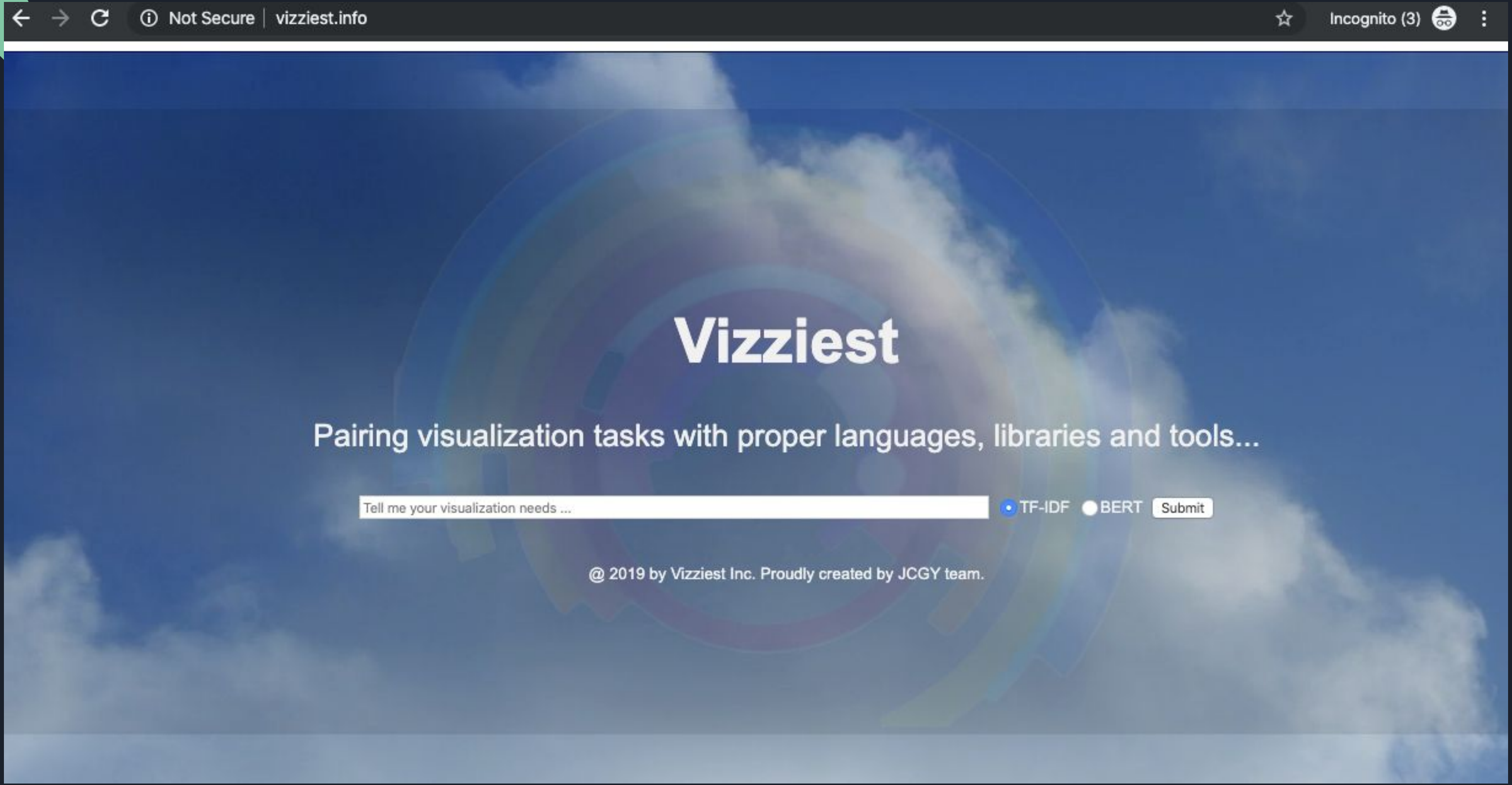
- **Our Mission**
  - **Vizziest** takes the time, frustration, and guesswork out of finding actionable guidance for creating the data visualization that best meets the user’s requirements.

- **Customer Pain Points**
  - **20 hours** spent per year researching solutions

- **Our Concept**
  - NLP models + filtered public data = a knowledge tool for creating visualizations

- **Future Potential**
  - Create similar specialized knowledge tools for other problem domains

Demo (http://vizziest.info)



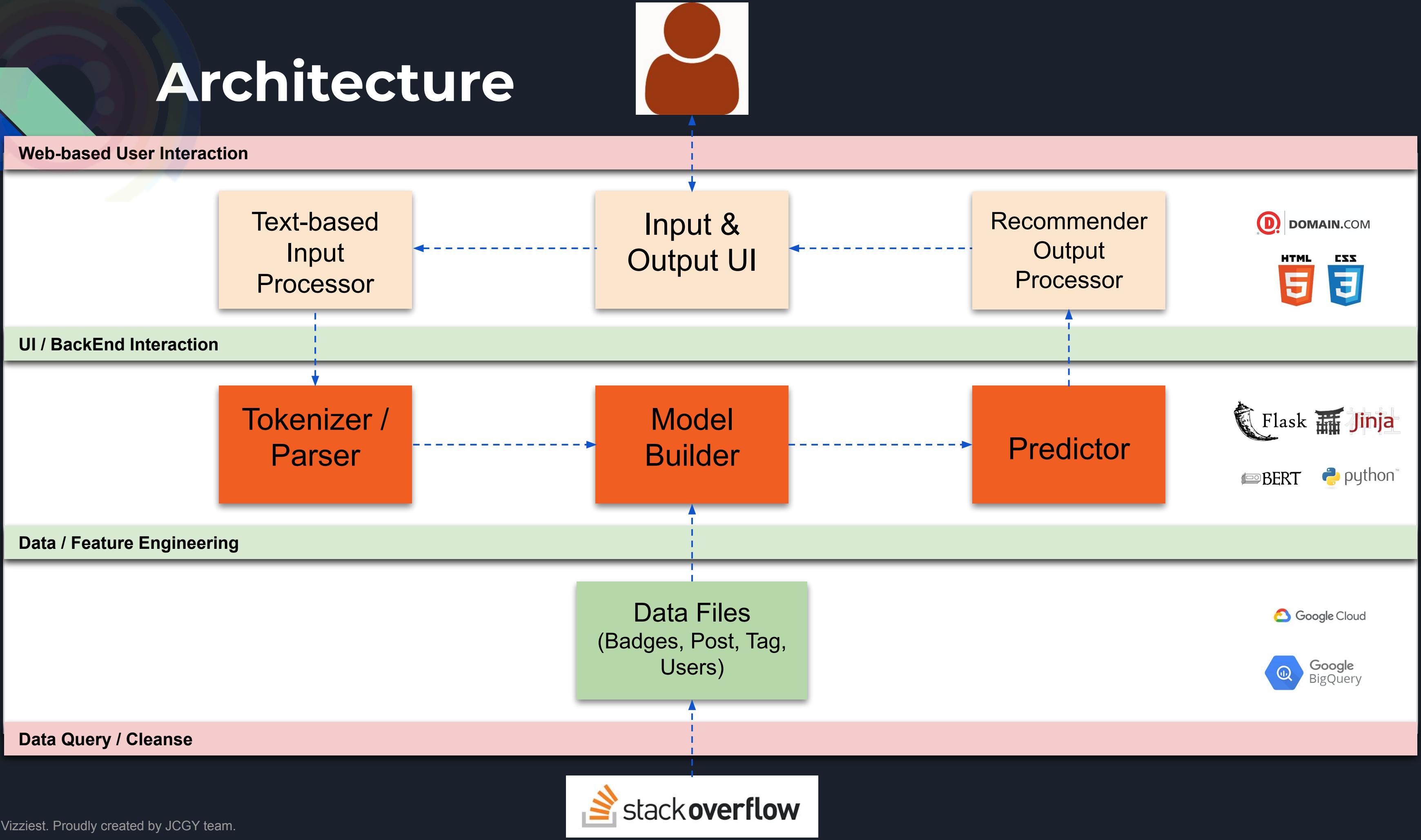
Vizziest

Pairing visualization tasks with proper languages, libraries and tools...

Tell me your visualization needs ...   ● TF-IDF ○ BERT Submit

@ 2019 by Vizziest Inc. Proudly created by JCGY team.

Architecture



How Vizziest works

- **User keys in a question**

- **Approach A**
  - **Various embedding approaches to vectorize questions:**
    - TF-IDF
    - Word to Vector
    - Sentence to Vector
  - **Logistic Regression to recommend questions and answers**

- **Approach B**
  - **BERT to directly recommend answers**
    - Next sentence prediction

Approach A: Models and Analysis

- **Baseline**
  - Top answer received from using the Stack Overflow search bar (Elasticsearch)

- **TF-IDF**
  - Uni-gram, bi-gram and tri-gram
  - Text frequency to determine importance
  - Pros and Cons: fast but no linguistic contexts

- **Word to Vector and Sentence to Vector**
  - A pre-trained model based on Google News
  - Neural networks trained to predict words from their neighbors
  - Pros and Cons: linguistic contexts but limited to Google News data and longer time

Approach A: Model Evaluation

- **Method**
  - Surveyed 105 people through Mechanical Turk
  - Compared best recommended questions from Stack Overflow, TF-IDF, and Word / Sentence to Vector embeddings

- **Result**
  - 52% of respondents chose TF-IDF uni-gram
  - 31% of respondents chose Stack Overflow search engine
  - 10% of respondents chose word to vector
  - 7% of respondents chose sentence to vector

Approach B: BERT Model and Analysis

- **BERT, Bidirectional Encoder Representations from Transformers**
  - Aims for better semantic understanding of search terms
  - Transfer learning

- **We used a pretrain BERT model from Hugging Face (pytorch)**
  - We need two sentences, or “spans” for the NSP task
  - User’s input becomes one of the spans
  - Stack Overflow accepted answer bodies the second span

- **Assumption**
  - The question should be semantically closer to the accepted answer body than other bodies
  - This turned out to be an insufficient assumption

Approach B: Model Evaluation

- **Method**
  - Surveyed 100 people through Mechanical Turk
  - Compared best recommended answers from Stack Overflow, TF-IDF, and BERT

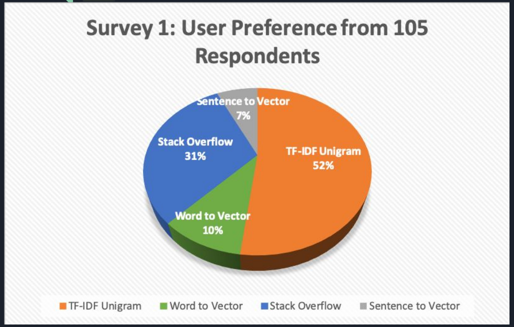
- **Result**
  - 62% of respondents chose TF-IDF uni-gram
  - 27% of respondents chose Stack Overflow search engine
  - 11% of respondents chose BERT

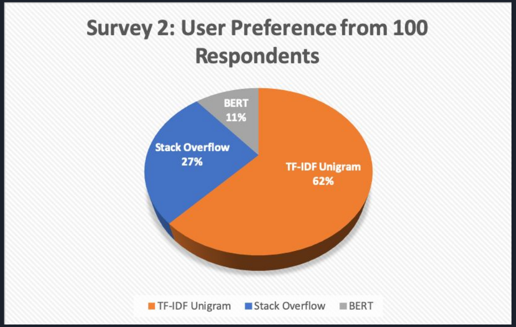
Approach B: Evaluation Takeaways

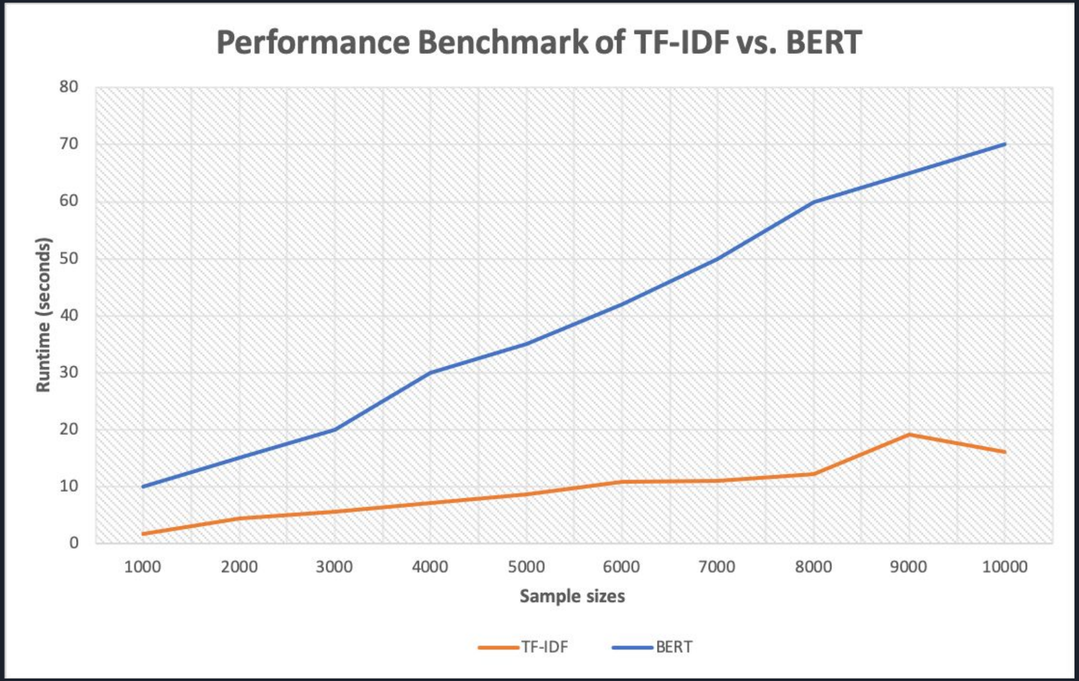
- **Overall**
  - For search applications, more traditional models, like TF-IDF offers better performance at lower cost than BERT

- **Performance**
  - Relevance is tough to measure on just the documents alone
  - Faster response times might be achieved with more GPUs
  - But, scales linearly with number of samples

User Preference and Performance Benchmark







Lessons Learned

- Careful filtering to create corpus is essential
- Leverage cloud offerings
- Open-ended search box alone may not be enough to capture user context
- Stack Overflow language is very different from normal speech and newspaper articles

Future Enhancements

- Add viz-relevant data from other sources
- Create manually-labelled “ground truth” dataset
- Incorporate user behavior into results evaluation
- Try hybrid models: TF-IDF + BERT Q&A
- More BERT fine-tuning

Q & A

Vizziest® = Making **Vis**ualization Ea**siest** for Everyone

Try it out!

http://vizziest.info

Capstone Project Gallery:

https://www.ischool.berkeley.edu/projects/2019/vizziest-making-visualization-easiest-everyone

Prior Work

Silva, Rodrigo F.g., et al. “Recommending Comprehensive Solutions for Programming Tasks by Mining Crowd Knowledge.” *2019 IEEE/ACM 27th International Conference on Program Comprehension (ICPC)*, 2019, doi:10.1109/icpc.2019.00054. (TF-IDF to search Stack Overflow for relevant answers, enhancing with API-related scores, and use final model over enhanced corpus to add explanations to final recommendations)

Anderson, Ashton, et al. “Discovering Value from Community Activity on Focused Question Answering Sites.” *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 12*, 12 Aug. 2012, doi:10.1145/2339530.2339665. (Predicting the lasting value of posts on crowdsourced knowledge sites such as Stack Overflow based on community and process dynamics features such as amount of answer and comment activity, speed of responses, answerer reputation, etc. Also predicts which questions need more answers.)

Wong, Edmund, et al. “AutoComment: Mining Question and Answer Sites for Automatic Comment Generation.” *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2013, doi:10.1109/ase.2013.6693113. (Highly specialized tool to identify code segments and create code comments based on Stack Overflow posts. Does filtering of posts to create initial corpus.)

Prior Work

Alreshedy, Kamel, et al. “Predicting the Programming Language of Questions and Snippets of StackOverflow Using Natural Language Processing.” 21 Sept. 2018, evaluatarXiv: 1809.07954v1 [cs.SE]. (Predict programming language of Stack Overflow posts using TF-IDF and Random Forest and XGBoost classifiers.)

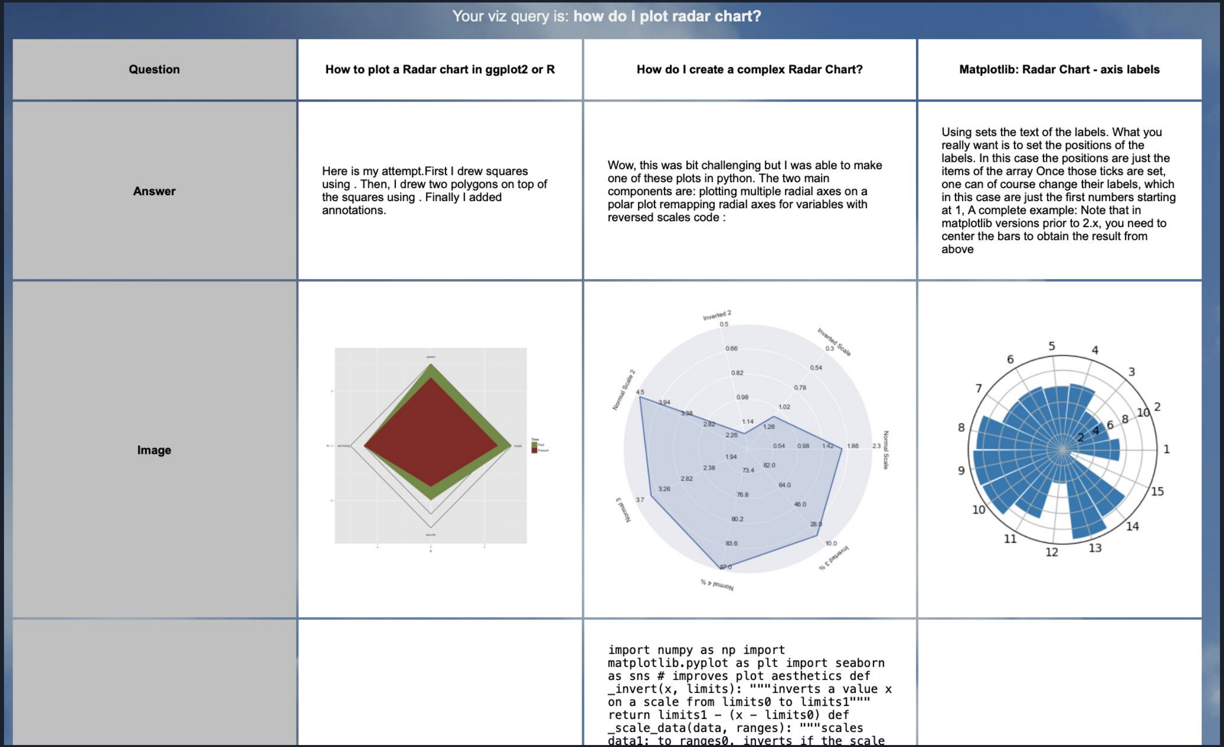
Devlin, Jacob, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Arxiv.org. Retrieved from https://arxiv.org/pdf/1810.04805.pdf

Lee, Ceshine. News Topic Similarity Measure using Pretrained BERT Model. Retrieved from https://medium.com/the-artificial-impostor/news-topic-similarity-measure-using-pretrained-bert-model-1dbfe6a66f1d

BERT by Hugging Face, Transformers, State-of-the-art Natural Language Processing for TensorFlow 2.0 and PyTorch. Retrieved from https://github.com/huggingface/transformers

Nayak, Pandu. Understanding searches better than ever before. Retrieved from https://www.blog.google/products/search/search-language-understanding-bert

TF-IDF Demo Results

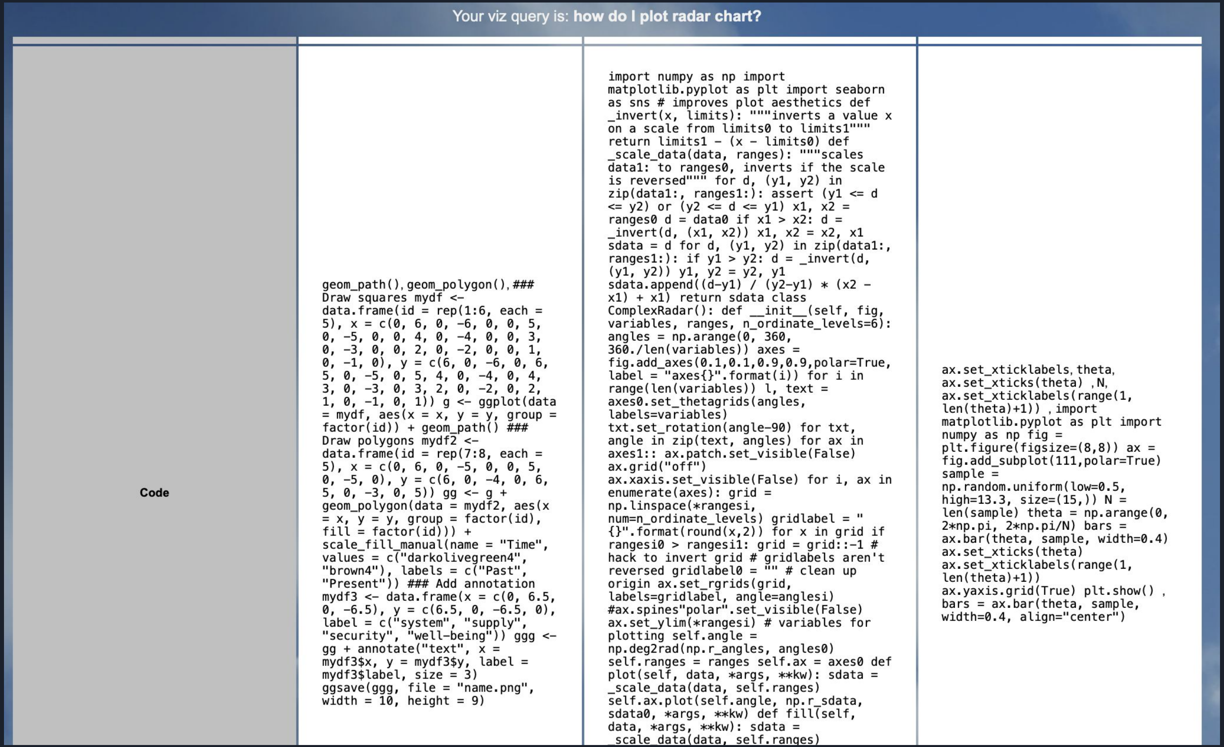


Your viz query is: **how do I plot radar chart?**

Question	How to plot a Radar chart in ggplot2 or R	How do I create a complex Radar Chart?	Matplotlib: Radar Chart - axis labels
Answer	Here is my attempt.First I drew squares using . Then, I drew two polygons on top of the squares using . Finally I added annotations.	Wow, this was bit challenging but I was able to make one of these plots in python. The two main components are: plotting multiple radial axes on a polar plot remapping radial axes for variables with reversed scales code :	Using sets the text of the labels. What you really want is to set the positions of the labels. In this case the positions are just the items of the array Once those ticks are set, one can of course change their labels, which in this case are just the first numbers starting at 1, A complete example: Note that in matplotlib versions prior to 2.x, you need to center the bars to obtain the result from above
Image			
		import numpy as np import matplotlib.pyplot as plt import seaborn as sns # improves plot aesthetics def _invert(x, limits): """inverts a value x on a scale from limits0 to limits1""" return limits1 - (x - limits0) def _scale_data(data, ranges): """scales data1: to ranges0, inverts if the scale	

@ 2019 by Vizziest. Proudly created by JCGY team.

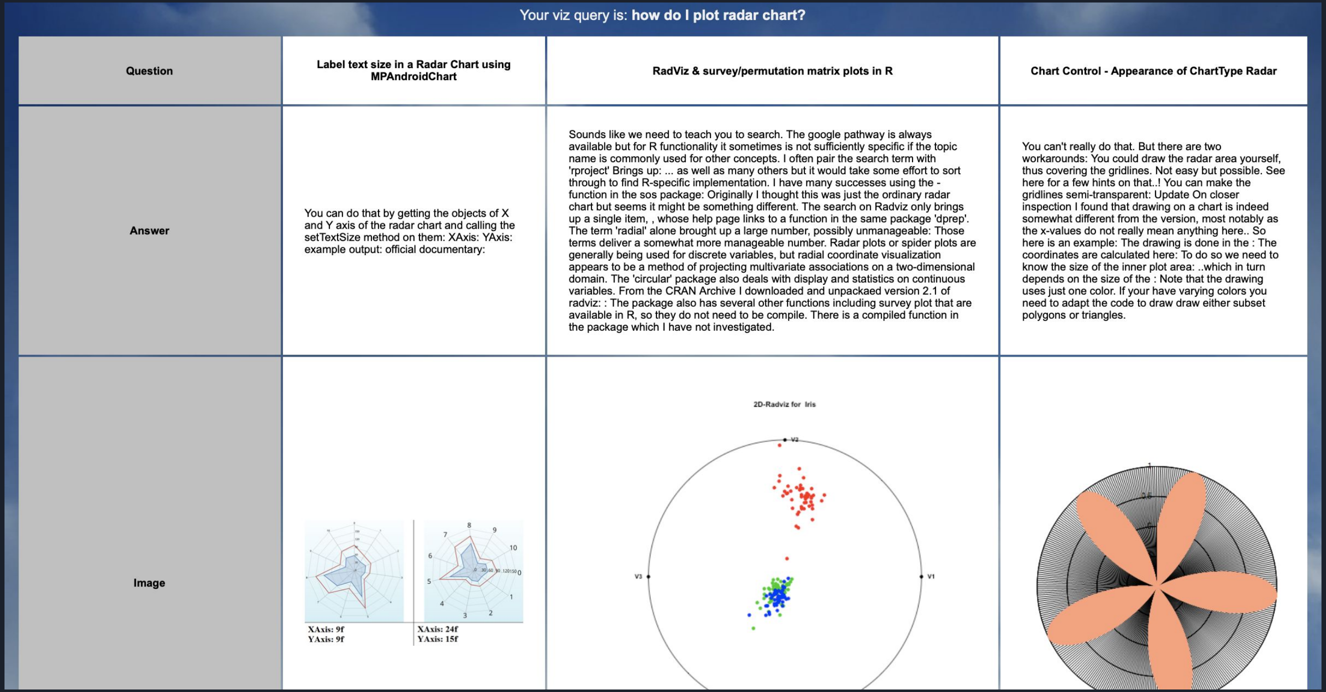
TF-IDF Demo Results



Your viz query is: **how do I plot radar chart?**

Code	geom_path(), geom_polygon(), ### Draw squares mydf <- data.frame(id = rep(1:6, each = 5), x = c(0, 6, 0, -6, 0, 0, 5, 0, -5, 0, 0, 4, 0, -4, 0, 0, 3, 0, -3, 0, 0, 2, 0, -2, 0, 0, 1, 0, -1, 0), y = c(6, 0, -6, 0, 6, 5, 0, -5, 0, 5, 4, 0, -4, 0, 4, 3, 0, -3, 0, 3, 2, 0, -2, 0, 2, 1, 0, -1, 0, 1)) g <- ggplot(data = mydf, aes(x = x, y = y, group = factor(id)) + geom_path() ### Draw polygons mydf2 <- data.frame(id = rep(7:8, each = 5), x = c(0, 6, 0, -5, 0, 0, 5, 0, -5, 0), y = c(6, 0, -4, 0, 6, 5, 0, -3, 0, 5)) gg <- g + geom_polygon(data = mydf2, aes(x = x, y = y, group = factor(id), fill = factor(id))) + scale_fill_manual(name = "Time", values = c("darkolivegreen4", "brown4"), labels = c("Past", "Present")) ### Add annotation mydf3 <- data.frame(x = c(0, 6.5, 0, -6.5), y = c(6.5, 0, -6.5, 0), label = c("system", "supply", "security", "well-being")) ggg <- gg + annotate("text", x = mydf3$x, y = mydf3$y, label = mydf3$label, size = 3) ggsave(ggg, file = "name.png", width = 10, height = 9)	import numpy as np import matplotlib.pyplot as plt import seaborn as sns # improves plot aesthetics def _invert(x, limits): """inverts a value x on a scale from limits0 to limits1""" return limits1 - (x - limits0) def _scale_data(data, ranges): """scales data1: to ranges0, inverts if the scale is reversed""" for d, (y1, y2) in zip(data1:, ranges1:): assert (y1 <= d <= y2) or (y2 <= d <= y1) x1, x2 = ranges0 d = data0 if x1 > x2: d = _invert(d, (x1, x2)) x1, x2 = x2, x1 sdata = d for d, (y1, y2) in zip(data1:, ranges1:): if y1 > y2: d = _invert(d, (y1, y2)) y1, y2 = y2, y1 sdata.append((d-y1) / (y2-y1) * (x2 - x1) + x1) return sdata class ComplexRadar(): def __init__(self, fig, variables, ranges, n_ordinate_levels=6): angles = np.arange(0, 360, 360./len(variables)) axes = fig.add_axes(0.1,0.1,0.9,0.9,polar=True, label = "axes{}".format(i)) for i in range(len(variables)) l, text = axes0.set_thetagrids(angles, labels=variables) txt.set_rotation(angle-90) for txt, angle in zip(text, angles) for ax in axes1:: ax.patch.set_visible(False) ax.grid("off") ax.xaxis.set_visible(False) for i, ax in enumerate(axes): grid = np.linspace(*rangesi, num=n_ordinate_levels) gridlabel = "{}".format(round(x,2)) for x in grid if rangesi0 > rangesi1: grid = grid::-1 # hack to invert grid # gridlabels aren't reversed gridlabel0 = "" # clean up origin ax.set_rgrids(grid, labels=gridlabel, angle=anglesi) #ax.spines"polar".set_visible(False) ax.set_ylim(*rangesi) # variables for plotting self.angle = np.deg2rad(np.r_angles, angles0) self.ranges = ranges self.ax = axes0 def plot(self, data, *args, **kw): sdata = _scale_data(data, self.ranges) self.ax.plot(self.angle, np.r_sdata, sdata0, *args, **kw) def fill(self, data, *args, **kw): sdata = _scale_data(data, self.ranges)	ax.set_xticklabels, theta, ax.set_xticks(theta) , N, ax.set_xticklabels(range(1, len(theta)+1)) , import matplotlib.pyplot as plt import numpy as np fig = plt.figure(figsize=(8,8)) ax = fig.add_subplot(111,polar=True) sample = np.random.uniform(low=0.5, high=13.3, size=(15,)) N = len(sample) theta = np.arange(0, 2*np.pi, 2*np.pi/N) bars = ax.bar(theta, sample, width=0.4) ax.set_xticks(theta) ax.set_xticklabels(range(1, len(theta)+1)) ax.yaxis.grid(True) plt.show() , bars = ax.bar(theta, sample, width=0.4, align="center")

BERT Demo Results

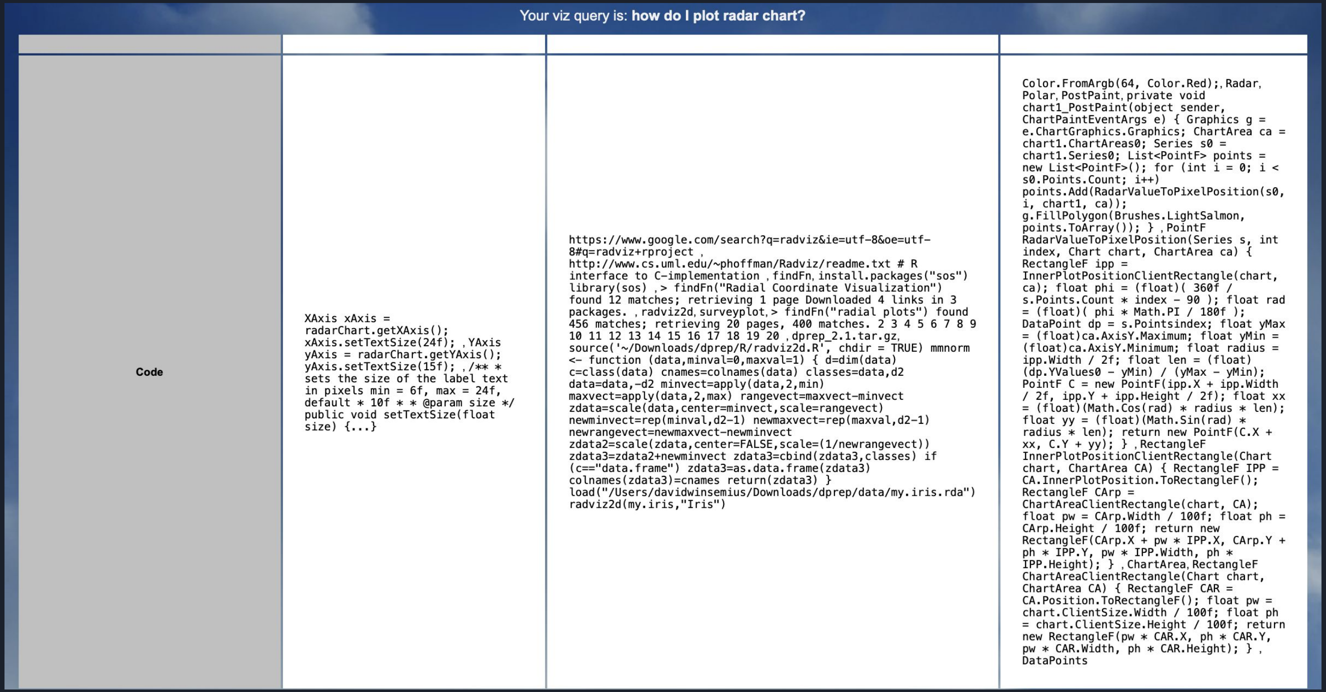


Your viz query is: **how do I plot radar chart?**

Question	Label text size in a Radar Chart using MPAndroidChart	RadViz & survey/permutation matrix plots in R	Chart Control - Appearance of ChartType Radar
Answer	You can do that by getting the objects of X and Y axis of the radar chart and calling the setTextSize method on them: XAxis: YAxis: example output: official documentary:	Sounds like we need to teach you to search. The google pathway is always available but for R functionality it sometimes is not sufficiently specific if the topic name is commonly used for other concepts. I often pair the search term with 'rproject' Brings up: ... as well as many others but it would take some effort to sort through to find R-specific implementation. I have many successes using the - function in the sos package: Originally I thought this was just the ordinary radar chart but seems it might be something different. The search on Radviz only brings up a single item, , whose help page links to a function in the same package 'dprep'. The term 'radial' alone brought up a large number, possibly unmanageable: Those terms deliver a somewhat more manageable number. Radar plots or spider plots are generally being used for discrete variables, but radial coordinate visualization appears to be a method of projecting multivariate associations on a two-dimensional domain. The 'circular' package also deals with display and statistics on continuous variables. From the CRAN Archive I downloaded and unpackaed version 2.1 of radviz: : The package also has several other functions including survey plot that are available in R, so they do not need to be compile. There is a compiled function in the package which I have not investigated.	You can't really do that. But there are two workarounds: You could draw the radar area yourself, thus covering the gridlines. Not easy but possible. See here for a few hints on that..! You can make the gridlines semi-transparent: Update On closer inspection I found that drawing on a chart is indeed somewhat different from the version, most notably as the x-values do not really mean anything here.. So here is an example: The drawing is done in the : The coordinates are calculated here: To do so we need to know the size of the inner plot area: ..which in turn depends on the size of the : Note that the drawing uses just one color. If your have varying colors you need to adapt the code to draw draw either subset polygons or triangles.
Image	XAxis: 9f YAxis: 9f XAxis: 24f YAxis: 15f	2D-Radviz for Iris	

@ 2019 by Vizziest. Proudly created by JCGY team.

BERT Demo Results



Your viz query is: **how do I plot radar chart?**

Code	XAxis xAxis = radarChart.getXAxis(); xAxis.setTextSize(24f); , YAxis yAxis = radarChart.getYAxis(); yAxis.setTextSize(15f); , /** * sets the size of the label text in pixels min = 6f, max = 24f, default * 10f * * @param size */ public void setTextSize(float size) {...}	https://www.google.com/search?q=radviz&ie=utf-8&oe=utf-8#q=radviz+rproject , http://www.cs.uml.edu/~phoffman/Radviz/readme.txt # R interface to C-implementation , findFn, install.packages("sos") library(sos) , > findFn("Radial Coordinate Visualization") found 12 matches; retrieving 1 page Downloaded 4 links in 3 packages. , radviz2d, surveyplot, > findFn("radial plots") found 456 matches; retrieving 20 pages, 400 matches. 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 , dprep_2.1.tar.gz, source('~/Downloads/dprep/R/radviz2d.R', chdir = TRUE) mmnorm <- function (data,minval=0,maxval=1) { d=dim(data) c=class(data) cnames=colnames(data) classes=data,d2 data=data,-d2 minvect=apply(data,2,min) maxvect=apply(data,2,max) rangevect=maxvect-minvect zdata=scale(data,center=minvect,scale=rangevect) newminvect=rep(minval,d2-1) newmaxvect=rep(maxval,d2-1) newrangevect=newmaxvect-newminvect zdata2=scale(zdata,center=FALSE,scale=(1/newrangevect)) zdata3=zdata2+newminvect zdata3=cbind(zdata3,classes) if (c=="data.frame") zdata3=as.data.frame(zdata3) colnames(zdata3)=cnames return(zdata3) } load("/Users/davidwinsemius/Downloads/dprep/data/my.iris.rda") radviz2d(my.iris,"Iris")	Color.FromArgb(64, Color.Red);, Radar, Polar, PostPaint, private void chart1_PostPaint(object sender, ChartPaintEventArgs e) { Graphics g = e.ChartGraphics.Graphics; ChartArea ca = chart1.ChartAreas0; Series s0 = chart1.Series0; List<PointF> points = new List<PointF>(); for (int i = 0; i < s0.Points.Count; i++) points.Add(RadarValueToPixelPosition(s0, i, chart1, ca)); g.FillPolygon(Brushes.LightSalmon, points.ToArray()); } , PointF RadarValueToPixelPosition(Series s, int index, Chart chart, ChartArea ca) { RectangleF ipp = InnerPlotPositionClientRectangle(chart, ca); float phi = (float)(360f / s.Points.Count * index - 90); float rad = (float)(phi * Math.PI / 180f); DataPoint dp = s.Pointsindex; float yMax = (float)ca.AxisY.Maximum; float yMin = (float)ca.AxisY.Minimum; float radius = ipp.Width / 2f; float len = (float)(dp.YValues0 - yMin) / (yMax - yMin); PointF C = new PointF(ipp.X + ipp.Width / 2f, ipp.Y + ipp.Height / 2f); float xx = (float)(Math.Cos(rad) * radius * len); float yy = (float)(Math.Sin(rad) * radius * len); return new PointF(C.X + xx, C.Y + yy); } , RectangleF InnerPlotPositionClientRectangle(Chart chart, ChartArea CA) { RectangleF IPP = CA.InnerPlotPosition.ToRectangleF(); RectangleF CArp = ChartAreaClientRectangle(chart, CA); float pw = CArp.Width / 100f; float ph = CArp.Height / 100f; return new RectangleF(CArp.X + pw * IPP.X, CArp.Y + ph * IPP.Y, pw * IPP.Width, ph * IPP.Height); } , ChartArea, RectangleF ChartAreaClientRectangle(Chart chart, ChartArea CA) { RectangleF CAR = CA.Position.ToRectangleF(); float pw = chart.ClientSize.Width / 100f; float ph = chart.ClientSize.Height / 100f; return new RectangleF(pw * CAR.X, ph * CAR.Y, pw * CAR.Width, ph * CAR.Height); } , DataPoints