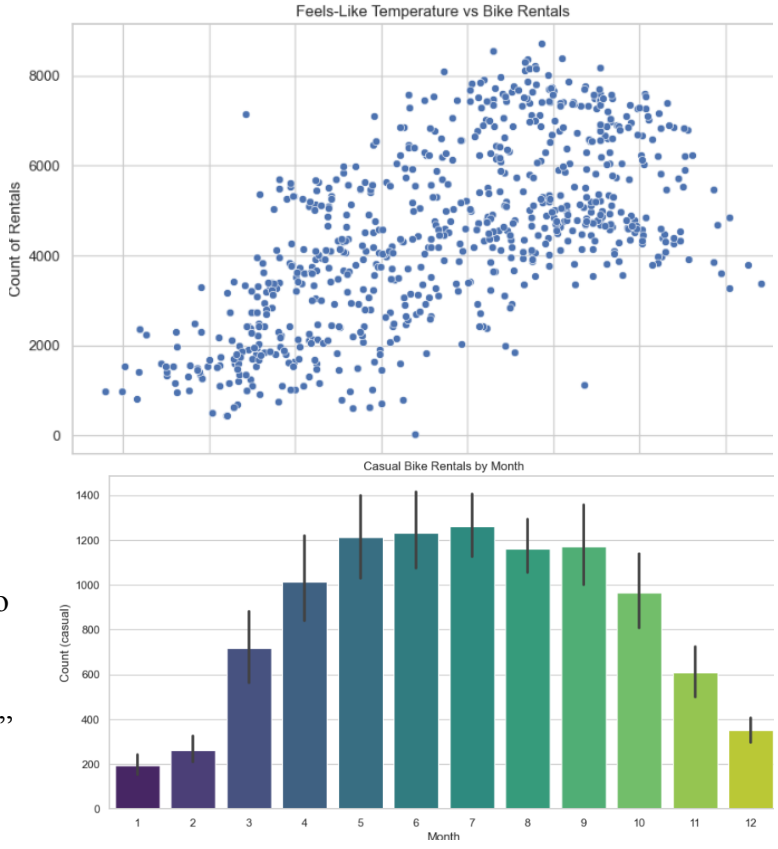
Bike Sharing Daily Demand Prediction

The start of our project was rough for us. We first planned on doing World Data, where we would predict immigration trends using news headings. We ended up running into a problem of collecting the data, it was impossible to collect world data policies regarding (attaining the data) (This was one of our 2 biggest mistakes from this project, you should always have data). We only had one option, which was to start a new project all over. The project we decided to do was Bike Sharing, the project's entire purpose was to be able to predict how many bikes are demanded on a day-to-day basis. The problem originally stemmed from one of our group member's friends who needed an algorithm to predict how many bikes are demanded on a day-to-day basis for the business not to have any loss, this is due to the high level of homelessness issues that have been rising in SF and Oakland Area. (This person puts out bikes for the day and collects them at the end of the day, so this something for the Bay-Area use)

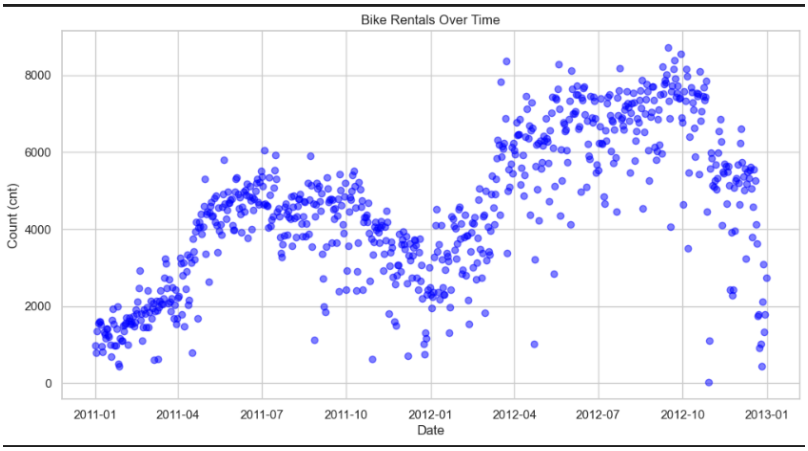
From our original mistake, we learned that we should attain the data before we submit our new proposal. So for our Data collection, we gathered data from various sources, just in case one of the "source" data wasn't good enough. One of our sources was Seoul Bike Data, which stems from the Kaggle competition on Bike Sharing Prediction (We **never** used this data but had it in case any of our other data didn't work out, also was very small, only 2 years' worth of data). We had data from Washington (state) assumed to be collected from Seattle. This data was also very small, only two years worth, and cleaned by UCI. We used this data to build our base models. And test our theory of what variables we should use for our big final project. And our last source, which our entire project was predicted and built upon was that of LA metro Bike Data. We also had LYFT data, the issue with this data was that it wasn't city-specific, and we needed to request the data, which was the same issue as our first project, needing access to download it (so we **never** used it).



Using our small Data set from Seattle, Washington which was from January of 2011 to December of 2012, exactly two years worth of data. There was very minimal data cleaning on this Data, however, we did a lot of EDA to test which variables were important and which weren't. So the variables we categorized into two categories, one was "Bike variables" which were "cnt, casual, registered, holiday, year, month, holiday, weekday, working day"

and the other was “Weather variables” which consisted of “season, weather, temp, atemp, hum, and wind speed”. We used all the variables to predict “cnt”, and we noticed that all the variables had either a positive or negative correlation when predicting the count of bikes. In Figure 1 & 2, you can see there is a correlation between Feels like temperature and Bike Rentals, all the weather variables followed similar patterns, whether it was a positive or negative correlation, but they tend to show that lower or higher values of their respective variables affected the amount of bike rentals per day. In a similar fashion as did the “bike variables”, if you see the months or group them by season, summer has the highest bike rental per day and lower bike rental per day during winter months.

After messing with EDA, we were ready to build our models. First, we decided which models we needed and which ones would be good. We assumed that the best model would be Random Forest, simply because of how the data is structured (categorical data variables and it building up on the,). The models we decided not to use or get rid of from scratch were logistic regression, decision Tree classifier, and Linear discrimination analysis, as they all are models to predict for classification, which isn’t something we are doing. Also, we did a random split for all our models, it seemed appropriate to do so since our data was time-series. For example the data for 2011 and 2012 in the figure, you can see how the trends go up and down but they are the same trend going up and down but amplified from 2011 to 2012 as business grew more in one year.



The first model we decided to build was linear regression, we were just curious how good it would be, and it would give insight to us very easily as that's the model you can change and adapt very quickly. We dedicated ourselves to building our linear model through the “statsmodel” package, this was simply done for us to see that summary table (I forget the name of this thing). This was for us to see which variables weren't good, through their coefficient value and p-values. We were looking for large coefficients and small p-values to determine which variables to keep. We noticed that they all seem important which is good news to us, and did VIF to check for variance, all

OLS Regression Results

Dep. Variable:	cnt	R-squared:	0.800
Model:	OLS	Adj. R-squared:	0.797
Method:	Least Squares	F-statistic:	261.9
Date:	Mon, 04 Dec 2023	Prob (F-statistic):	7.80e-243
Time:	20:29:26	Log-Likelihood:	-5981.0
No. Observations:	731	AIC:	1.199e+04
Df Residuals:	719	BIC:	1.204e+04
Df Model:	11		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1469.0031	240.218	6.115	0.000	997.390	1940.616
season	509.7752	54.757	9.310	0.000	402.272	617.278
yr	2040.7034	65.185	31.306	0.000	1912.727	2168.680
mnth	-38.9796	17.079	-2.282	0.023	-72.510	-5.449
holiday	-518.9919	201.040	-2.582	0.010	-913.688	-124.296
weekday	69.0622	16.299	4.237	0.000	37.063	101.061
workingday	120.3570	72.007	1.671	0.095	-21.013	261.727
weathersit	-610.9870	78.363	-7.797	0.000	-764.835	-457.139
temp	2028.9161	1403.671	1.445	0.149	-726.867	4784.699
atemp	3573.2743	1589.389	2.248	0.025	452.877	6693.671
hum	-1018.8616	313.995	-3.245	0.001	-1635.318	-402.405

...

variables were low so another reason for us to keep them. Our adjusted r-squared was almost 0.80. Indicating that we had a perfect model here. Our r-squared score on the test set was 0.83, indicating we didn't overfit our model to our train model as it had increased on the test set. We decided to test our theory of "weather variable vs Bike variables". We noticed that they don't predict well independently of one another, indicating that both variables (bike and weather) mattered a lot. We tested this theory by making linear regression, which predicts starting only day, month, and year. It only predicted an r-square of 0.41 which was bad, and then we did an r-square by adding temp and hum it went all the way up to 0.75 which was good, and then adding all the variables you get 0.797. Indicating both variables mattered a lot.

Our next model was that of LTSM, we were just curious how good a neural network would be for model building. This model is supposed to be good with time series-based data, so its a no-brainer to pick this model. The model we built at first was using all the variables similar to that of linear regression and just ran it. We noticed our r-squared was negative, this seemed weird to us because the data was really good, it has patterns so why is LTSM predicting a negative r-squared. We did some research and it turns out that if your data is really small or doesn't have a lot of points it will predict badly, the solution to this is to have bigger data or a lot of epochs. So we started with 100 epochs and still negative r-squared, and then we ran 1000 epochs. It gave us an r-squared of 0.88 on the test set data, which is better than linear regression which we had assumed it should be able to do. We messed with the epochs value, putting at 500, where it gave us r-square still, and 10000, it gave us the same r-squared, so we decided to stay with 1000 epochs as it was giving us the best results in terms of adjusted r-squared on test. We did take into consideration that we might be over-training our data, as 1000 epochs is a lot, so we decided to through "Earlystopping" which helps the model from over-fitting. We still had the same r-squared, and then we started to do boosting, bootstrapping, and cross-validation to make sure no over-fitting existed in the data. We noticed the value of r-squared would drop but it only dropped down to 0.86 with all three added on to LTSM. Indicating our model was good, but there is some over-fitting going on. Also, we did the comparison test here as well of the variables of "Bike vs Weather", it was more interesting we got a value of 2188 for without weather, which was using only bike data, and just weather got 1090 and the combination of both got 2626. The number is supposed to be close to 2729, we were trying to predict the bike count for one specific day of 12-30-12. We learned that Bike data here matters a lot, however, that's probably because most of the data was time series for "bike variables". Here we concluded that it wasn't worth dropping any variables after this for any of our models, having all the variables is a good thing, and helps with r-squared and they don't have high variance.

Our next and Final Model is that of Random Forest, which was the model we assumed would predict the best, but we built it last because we wanted to see other models predicting and comparing them. Our assumption and theory were correct, random forest did the best prediction, it predicted around 0.87 on the test set, which is .1 less than LTSM however, When we added boosting, bootstrapping, and cross-validation to the model we noticed that it kept increasing its adjusted r-square. It increased to 0.90. That indicated that it wasn't getting over-fitted.

Now being done with our models with small data and knowing which variables were important we started to work on the large data set. Which was the LA metro- dataset. This entire thing was where our most time was spent, downloading, combining the different CSV and cleaning, and then adding on weather API as it didn't have the weather variables. So first we tried to attain the weather API, we built our script for weather API, which gives you weather variables for giving location, place, and time, which is what we needed. Now we made a script that went on the LA metro website and downloaded all the data. After downloading the data, it would combine all the files into CSV and save it as a CSV file which we can open and start to work on. This is where our issues started to rise. When we first did the models we noticed that there were a lot of NA values, we had to decide which values we should drop and which ones we shouldn't. So our solution was let's just plug in 0 for the NAN values and see how our model does, and then compare and contrast when removing the variables, we used linear regression as the LTSM model kept crashing (this was due to high vectors and too much memory overload on the cpus), and Random Forest would take more than 4 hours of run time. We get the results of 0.18 for the Linear regression model with NAN values replaced with 0 and an increase of adjusted r-square on test results of 0.227. So it was better to drop the values. However, we started noticing that it didn't make sense that our model was this bad given how large our dataset is, why is it that the adjusted r-square is so low when we had the same model for our other dataset. Something didn't seem right, so we started doing EDA, as it didn't seem like the model was doing good, as we predicted that the model should have done better. This is where we found the biggest problem, around line 54359 (index) the variables changed their date format, it was following the data form of mm/dd/yy to doing dd-mm-yy ss-mm-hr, and this was throwing our script off, and mismatching and matching data randomly with 0 values, which is was why we had such low r-squared values. It took us approximately 3 days to fix this issue of data fixing. After we got this fixed we ran Linear regression the model predicted 0.67 which isn't great. But we wanted to work and test Random Forest which is supposed to be our best model for this entire thing. We ran the model, and we got 0.999997 with an MSE of 21, which is good. We did more testing to make sure our model wasn't just precut high adjusted r-square, it was still the same after applying boosting and cross-validation. We did some more analysis through "Feature importances" removing features that were important to predicting that high r-squared, and the value would drop down significantly to 0.78 after removing the most important variables. So we concluded that our Random Forest model indeed isn't overfitting after applying cross-validation and removing variables. It predicted close to that 0.99 with a very low MSE. In conclusion, Random Forest worked the best, and we were successful at predicting daily bike usage. That weather plays an important role when predicting bikes per day as much as bike variables themselves, and combining them was a good thing.

Data used Websites:

- https://bikeshare.metro.net/about/data/
- https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset
- All the variables that we described can be found in the UCI, education, read me page when you download it. It explains what each variable is. Similarly so does the metro data.

Link to our Code and Files:

- https://drive.google.com/drive/folders/1EiJSmj52UsB33Zip7dv2CKjRFYq89WTV?usp=sharing
- Please read the text file titled, “Instructions_how_to_Run_Code to run the code if you want to replicate the results. All the code should compile without error as long as you set up the working directory correctly, and just run the code.